J.J.MEYER

3D et VRAI RELIEF

IMAGES DE SYNTHESE

initiation
CAO-DAO
sur tous micros



ÉDITIONS RADIO

J. BENARD

REPERTOIRE MONDIAL des BASIC

EQUIVALENCES.TRANSPOSITION.SIGNIFICATION

MICRO · MOTS CLÉS · UTILISATION · MULTI BASIC

EDITIONS RADIO

J.C. GUILLEMOT

INITIATION PASCAL

INITIATION aux FICHIERS BASIC

J. BENARD

EOF · GET · FIELD · CLOSE · PUT · INPUT#

H. LILEN

INITIATION BASIC

LIST ON GOSUB POKE · INPUT RUN NEXT · NEW GO TO · CSAVE · RETURN

de nombreux ...ples pratiques ...asic Microsoft

H. LILEN

INITIATION BASIC

H. LILEN

8088 ASSEMBLEUR

IBM PC et compatibles

3 livres

COURS D'INITIATION
MANUEL DE L'UTILISATEUR
GUIDE PRATIQUE





Éditions Radio

9, RUE JACOB - 75006 PARIS

TEL. 43.29.63.70

3D
et
VRAI RELIEF

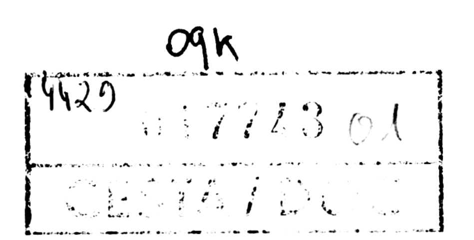
IMAGES DE SYNTHESE

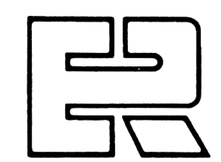
Ce livre est dédié à mon épouse, qui, pendant de longues soirées, a préparé le manuscrit de cet ouvrage.

J.J.MEYER

3D et VRAI RELIEF

IMAGES DE SYNTHESE





ÉDITIONS RADIO
9, RUE JACOB - 75006 PARIS
TEL. 43.29.63.70

Imprimé en France par Berger-Levrault, Nancy

Dépôt légal : septembre 1986
Éditeur nº 1023 - Imprimeur : 779006
I.S.B.N. 2 7091 0990 5

INTRODUCTION

Le monde de la micro-informatique connait en ce moment des mutations profondes. Les programmes de *jadis*, aux défilements d'écrans simplistes et dépouillés à l'extrême de présentation graphique sont-ils encore utilisés ? Que sont devenues les machines à 4 ou 7 K de mémoire vive, avec des systèmes d'exploitation squelettiques ?

De nos jours les micros affichent couramment avec suffisance des RAM de 128 K. Les sytèmes d'exploitation sont complets, d'emploi facile et certains utilitaires permettent même d'en rendre l'utilisation particulièrement agréable.

Quant aux programmes, il suffit d'utiliser les plus célèbres pour s'apercevoir que les fenêtres et les graphismes sont devenus quasiment indispensables. Les histogrammes, pour leur part, arborent souvent fièrement la troisième dimension.

Même les micros dits *familiaux* offrent tous en standard la haute résolution et la couleur, 64 K de RAM et une vitesse C.P.U. de 4 Mhz.

Au niveau des dessins et des graphiques, le progrès a suivi le même chemin. Les personnages ont abandonné leur apparence cubique sur un fond désespérément noir, pour un aspect plus humain utilisant même des paysages en couleur et en trois dimensions. Dans le domaine du dessin graphique les rosaces en deux dimensions sont devenues désuettes, les tracés tridimensionnels avec effacement des surfaces cachées les remplaçant avantageusement. Il est même possible de simuler des rotations de formes dans l'espace.

L'objet de ce livre est de sublimer ces techniques en poursuivant l'évolution selon les mêmes principes. En effet, mieux que la troisième dimension, voici maintenant le **relief**. Il convient de préciser qu'il s'agit réellement de relief et non pas d'une simple simulation.

Ainsi l'utilisateur aura réellement l'impression de voir un objet *sortir* de l'écran pour venir à sa rencontre. Il lui sera possible de situer spatialement les différentes parties de cet objet; ce qui lui permettra de percevoir la distance séparant les différents plans du dessin.

Mais avant de parvenir à ce résultat, il est impératif de bien saisir tous les concepts qui régissent les lois de la perception du relief. En effet, la vision humaine obéit à des mécanismes extrêmement complexes (physiques, chimiques, biologiques et psychologiques). Par contre c'est principalement notre cerveau qui *voit* et, de plus, opère des sélections. La complexité est encore augmentée par le fait que notre environnement dépend de la troisième dimension. De plus il convient de ne pas oublier que le cerveau est capable de synthétiser des images en relief à partir de simples photons frappant notre rétine.

Ceci implique énormément de problèmes à résoudre pour arriver à faire croire au cerveau que ce qu'il voit sur un écran cathodique est identique à ce qu'il verrait réellement dans l'espace.

Seuls des logiciels très spécifiques sont capables d'atteindre ce résultat, qu'il s'agisse de visualisations sur écran de micro-ordinateur, sur imprimante, sur table traçante ou par l'intermédiaire d'un *stéréoscope.*

Mais ce n'est pas suffisant, il faut aussi utiliser des techniques empruntées à la photographie, simples ou de plus en plus sophistiquées, selon les résultats recherchés. Ce n'est qu'à ce prix que le lecteur sera amené à réaliser des dessins en trois dimensions, voire même des animations, avec adjonction possible du relief.

Cependant ce livre serait incomplet s'il négligeait les innombrables possibilités que les dessins en trois dimensions et le relief sont capables d'apporter aux applications professionnelles.

Les domaines concernés sont aussi diversifiés que ceux de l'enseignement, de la gestion, ou du dessin industriel (C.A.O.). Des bibliothèques de formes peuvent apporter une aide intéressante à ce niveau. Si de plus le concepteur peut voir exactement son projet comme s'il était déjà réalisé, il ne s'agit plus d'une aide mais d'un outil indispensable.

Afin que chacun puisse utiliser aisément ces techniques, les différents programmes ont été écrits en **BASIC MICROSOFT** classique. S'ils sont dépouillés de toute présentation sophistiquée, c'est uniquement dans le but de permettre d'en saisir rapidement l'essentiel et de les rendre les plus universels possible. Seul un sous-programme nécessitera une adaptation pour chaque machine : celui du tracé en haute résolution. Mais cette adaptation sera très simple, elle se limitera pour l'essentiel au tracé de segments de droites, c'est-à-dire à l'équivalent de **HPLOT** sur **APPLE** ou **DRAW** sur d'autres matériels (voir en annexe les instructions spécifiques à chaque machine). Ainsi n'importe quel micro-ordinateur sera à même de les utiliser, pour peu qu'il dispose d'une haute résolution, d'une imprimante ou d'une table traçante. Pour le relief, il est conseillé, en plus, de disposer de la couleur. Enfin, pour ceux que la rapidité d'exécution préoccupe, un chapitre consacré au **TURBO PASCAL** permettra d'atteindre la vélocité nécessaire.

Le but de ce livre serait atteint si le lecteur poursuivait l'évolution citée plus haut, en développant lui-même ses propres logiciels, en les rendant plus performants et surtout en effectuant à son tour ses propres découvertes.

CHAPITRE 1

LA VISION HUMAINE

Les phénomènes naturels qui participent à l'élaboration de la vision de l'être humain sont extrêmement complexes. Pourtant nous n'en avons généralement pas conscience. Bien entendu chacun de nous conserve encore le vague souvenir de cours de Sciences Naturelles, dans lesquels il était question de globe occulaire ressemblant vaguement à un appareil photographique.

En fait, raisonner égocentriquement équivaut un peu à injurier la nature. Ce serait plutôt la machine qui tenterait d'imiter celle-ci, avec des résultats qui sont encore bien loin d'égaler les performances incroyables atteintes par l'œil.

Cependant, depuis quelques années, les progrès scientifiques ont énormément avancé. Ainsi, pour prendre un exemple, l'enseignement de la physique dans les écoles ne descend toujours pas plus loin que les protons, les neutrons et les électrons, au niveau de la description atomique; alors que l'on sait désormais que ces particules sont elles-mêmes composées de *quarks* et de *gluons.* Au niveau de la recherche scientifique, le progrès est tellement rapide que l'information officielle ne suit pas.

Le processus est identique en ce qui concerne la vision. Régulièrement de nouvelles découvertes viennent approfondir la connaissance en cette matière. Il est donc important de revoir le rôle de l'œil dans la capture des rayons lumineux, ainsi que le celui du cerveau lorsqu'il s'agit d'en interpréter les données.

1. L'ŒIL

Notre environnement baigne constamment dans un fourmillement d'ondes de toutes natures, mais nous avons coutume d'appeler « rayons lumineux » celles que notre œil est capable de discerner.

Rôle actif

A chaque instant, une multitude de grains de lumière (les *photons*), pénètre à l'intérieur de nos yeux. Mais le fond de l'œil n'a rien de comparable à la surface d'une simple pellicule photographique reflètant le monde extérieur. Son rôle est actif, car il est constitué de tissus cérébraux, lesquels se sont détachés du cerveau au moment où l'embryon humain se développait.

Nous sommes donc assez éloignés du simple rôle passif qui lui a longtemps été attribué: l'œil est une structure nerveuse qui capte une présence lumineuse, mais aussi qui l'amplifie. Et surtout il apparaît de plus en plus évident que sa fonction consiste principalement à opérer un certain tri de ces informations, pour les présenter au cerveau prêtes à être exploitées rapidement par ce dernier.

Cette propriété est très importante pour la suite de notre propos. Très souvent des études (intéressantes par ailleurs) montrent les limites de la perception humaine, ainsi que les « défauts » de notre œil. Mais s'agit-il vraiment de défauts ? L'œil a été conçu de manière à permettre au cerveau de situer spatialement le corps dans un contexte naturel, avec une recherche constante des performances.

Mécanisme de la vision

Dans ces conditions il n'est pas étonnant que certaines combinaisons soient capables de provoquer des interprétations erronées, ces situations ayant peu de chance de se réaliser normalement. Par contre il s'agit d'une véritable aubaine à notre niveau.

Le mécanisme de la vision est à la fois simple et complexe. Reportez vous aux figures 1.1 et 1.2, pour l'illustration de notre propos. Il est simple dans le principe de son cheminement: un photon a été réfléchi par un objet. Par hasard il traverse la cornée de l'œil, réussit à franchir la cible de la pupille, passe au travers du cristallin et poursuit son chemin au sein de l'*humeur vitrée*, pour finalement atteindre la *rétine.*

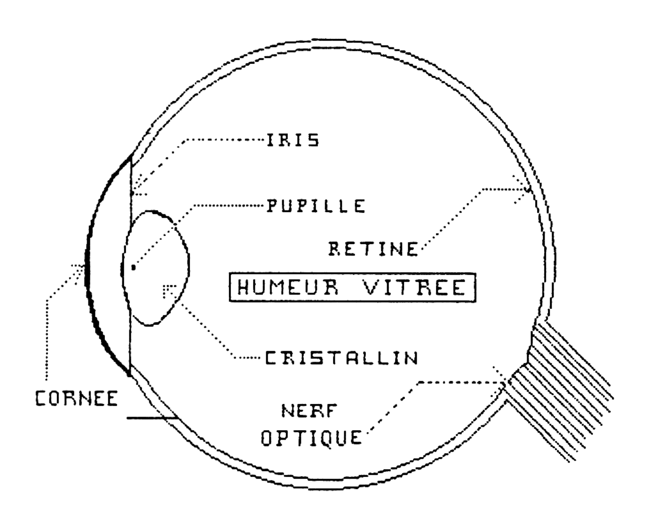


Figure 1.1 — Vue en coupe d'un œil.

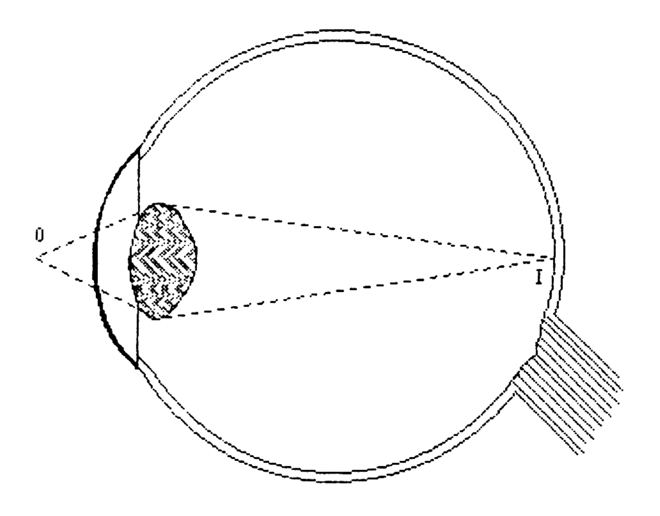


Figure 1.2 — Chemin emprunté par les photons à l'intérieur d'un œil.

A partir de ce moment commence l'élaboration de phénomènes complexes. Chacun sait que la rétine est tapissée de *cônes* et de *bâtonnets*. Leur tâche revêt la plus haute importance.

Mais voyons d'abord quelle est la nature physique des rayons lumineux.

2. LES RAYONS LUMINEUX

Dans la nature, tout n'est que vibrations: une onde électromagnétique est issue d'une vibration atomique. La température ne traduit que des différences de vibrations, pas une quantité mesurable: le *zéro absolu* (−273 degrés Celcius) correspond à une immobilité totale des atomes et plus les vibrations s'intensifient, plus la température (telle que nous la définissons), augmente.

Perception des rayons lumineux

Le *photon* est un grain d'énergie lumineuse associé à une onde électromagnétique. De sa longueur d'onde dépend la quantité d'énergie qu'il transporte. Or notre œil ne réagit qu'à une certaine bande de ces longueurs d'ondes: vers 380 nanomètres, il percevra une sentation de violet et autour de 750 nanomètres, la couleur sera située du côté du rouge. Une certaine bande de fréquences est détectée par les yeux, une autre par les oreilles et ainsi de suite. Pourtant il ne s'agit toujours que de la manifestation perceptible de vibrations d'atomes.

Il est donc important d'insister sur le fait que les couleurs et les sensations lumineuses, telles que nous les comprenons généralement, sont faussement interprétées. Que signifie « rayon lumineux »? Tout simplement la bande de fréquences que l'œil de l'homme est capable de percevoir. Mais alors comment appeler les rayons que d'autres animaux, pour leur part, percoivent? Les abeilles discernent des ultraviolets jusqu'à 310 nanomètres, tandis que les serpents détectent des infrarouges de 770 nanomètres, donc « voient » la chaleur! Encore une fois il ne faut pas tout ramener au niveau de l'homme. Voir signifie tout simplement être capable de réagir à une certaine bande de fréquences électromagnétiques avec plus ou moins d'acuité selon le niveau de performances de l'organe sensoriel détecteur.

Cônes

C'est par la diversité des *cônes*, que les couleurs sont perçues: certains cônes sont plus sensibles à la longueur d'onde se traduisant par une sensation de bleu, d'autres à celle correspondant au rouge et enfin les derniers à la fréquence du vert. Ceci explique, entre autres, la conception en mode **R.V.B.** des moniteurs couleur (Rouge Vert Bleu, ou **R.G.B.** pour les anglo-saxons); Si la répartition de l'ensemble des photons frappant simultanément la rétine est homogène, une sensation de blanc apparaîtra. En l'absence de tout photon, le noir dominera. Et selon les proportions des différentes énergies, tout le spectre de l'arc-en-ciel pourra être représenté.

Bâtonnets

Donc, autour de nous, tout est irrémédiablement noir. Les sensations de couleurs résultent simplement de la traduction des différentes énergies des photons en signaux « codés » sur trois niveaux. Mais ce sont les *bâtonnets* qui effectuent la conversion de l'énergie du photon en stimulation nerveuse, directement exploitable par le cerveau.

3. LA CHIMIE DU CERVEAU

Suite à ces considérations, il apparaît difficile de situer avec précision la frontière fonctionnelle entre l'œil et le cerveau. Le rôle des bâtonnets est assez spécifique. En effet, bien qu'incapables de nuancer les teintes comme les cônes, ils transmettent avec précision au cerveau la quantité de lumière que la rétine vient de recevoir. Ils sont tellement sensibles qu'ils peuvent réagir à l'impact d'un seul photon.

Ce choc a pour conséquence de modifier l'agencement des atomes, au niveau des molécules composant les petits disques empilés qui forment un bâtonnet. A partir de cet instant, une succession de réactions chimiques va s'enchaîner, afin de donner naissance à une impulsion qui se propagera le long du bâtonnet. Il est remarquable de constater que le bâtonnet n'est pas un simple récepteur passif, mais bien une cellule nerveuse à part entière, tout à fait comparable aux *neurones* composant le cerveau.

Puis la structure moléculaire du disque frappé par le photon se reconstitue, attendant un nouveau photon. Les disques sont remplacés régulièrement, par la base, à la fréquence d'un nouveau en moins d'une demie heure.

Mise en forme des informations

Bien entendu, toutes ces réactions ne se sont passées en réalité qu'en quelques millisecondes. On pourrait presque dire que le fond de l'œil n'est qu'un prolongement du cerveau, assez spécialisé il est vrai. Enfin le tout s'accompagne d'un phénomène multiplicateur, de l'ordre du millier de fois. Les performances sont telles, que des différences d'intensités lumineuses de un million sont parfaitement différenciées.

L'œil vient d'accomplir sa mission. Il a mis en forme les informations qu'il transmet maintenant au cerveau, c'est-à-dire la valeur de l'intensité du rayonnement perçu, sa composition au sein de la gamme des fréquences à prendre en considération, la moyenne des fréquences qui le composent et le tout élevé à un niveau exploitable.

Dès cet instant commence la partie de la vision dite *cérébrale.* Les couleurs des images reçues sont créées par le cerveau, alors qu'elles nous paraissent faire partie intégrante de l'objet que nous observons. La chimie du cerveau, capable de transformer de simples impulsions lumineuses en images composées de couleurs éclatantes, est encore mal connue. Du moins avec moins de précision que le mécanisme de la réception des rayons lumineux par l'œil.

Nous avons vu plus haut qu'un premier traitement était effectué, par l'œil lui-même. En effet, la rétine est obligée de regrouper les informations, car celles-ci sont fournies par 100 millions de bâtonnets et 20 millions de cônes, alors que « seulement » un million de *fibres* composent le nerf optique. Afin de limiter les regroupements d'informations, le nerf optique véhicule principalement celles en provenance des bâtonnets. Les cônes, pour leur part, sont le plus souvent « en prise directe » avec le cerveau.

Synthèse des données

La conséquence la plus importante de cette situation, c'est la synthèse des données provenant des bâtonnets connectés à une même fibre optique: seule la moyenne de milliers de photons reçus est prise en compte.

Il est remarquable de constater que, jusqu'à présent, les signaux étaient de forme *analogique* (comparaisons et additions). A partir du nerf optique, le traitement devient *logique*: les signaux, préparés sous forme de trains d'impulsions, sont acheminés vers le cerveau.

Chiasma optique

Puis, sous celui-ci, les deux nerfs optiques de nos yeux se rejoignent et se croisent à un emplacement appelé *chiasma* optique. C'est à cet endroit, comme nous l'étudierons plus loin, qu'un mécanisme se prépare, en vue d'assurer la perception de notre environnement en trois dimensions.

Mais en attendant, considérons l'arrivée d'un nerf optique dans le cerveau. L'entrée se situe au niveau du *cortex* visuel. A partir de cette zone, les informations diffusent dans plusieurs directions et à différents niveaux, comme l'ont démontré de récentes découvertes soviétiques. Cette reconnaissance des formes sera traitée au chapitre suivant, mais d'ores et déjà il est possible d'affirmer que le cortex visuel ne suffit pas à former des images cérébrales à partir de signaux visuels. D'autres zones du cerveau vont concourrir, par l'intermédiaire de leurs informations, à l'élaboration de ces images.

Néanmoins le rôle du cortex visuel est très important. Il est capable de transformer les stimulations reçues en bandes d'informations multidirectionnelles, afin

de pouvoir détecter un éventuel déplacement et lui donner une direction. Les coordonnées **X** et **Y** du champ visuel sont ainsi prises en compte, mais aussi les interconnexions en provenance de plusieurs cellules primaires. Ce qui fait que le traitement de l'information, véhiculée sous forme digitale, redevient *analogique* (comparaisons, sommations, analyses). De plus cette zone du cerveau commande, de façon inconsciente, les mouvements de l'œil et l'*accomodation*, par l'intermédiaire de muscles appropriés.

Perception des images

Enfin le cortex visuel procède aussi à l'élaboration des couleurs. Les signaux électriques reçus du nerf optique sont traduits. La sensibilité des cônes est 1000 fois inférieure à celle des bâtonnets, ce qui explique la perte de la notion de couleur pour les faibles intensités lumineuses. Par contre, les trois sortes de cônes permettent des comparaisons pour les stimuli reçus simultanément par plusieurs groupes de cônes, de sensibilité différente.

Les couleurs sont donc bien « inventées » par le cerveau ; l'œil ne reçoit que des radiations électromagnétiques, que le cerveau est même capable de corriger automatiquement. Par exemple les teintes perçues sont le plus souvent traduites en leurs véritables niveaux, malgré les variétés des différentes sources lumineuses environnantes, ce que ne peut effectuer une pellicule photographique. Ceci est le résultat d'un traitement de l'information complexe, sur plusieurs niveaux.

Ainsi, après une certaine période d'adaptation, l'œil et le cerveau sont capables de recréer un contexte modifié par un environnement trompeur. C'est un point très important pour la visualisation en relief par notre procédé: il suffira d'agencer d'une certaine façon des images en deux dimensions, pour avoir l'impression d'observer non seulement une image en trois dimensions, mais plus encore un *véritable relief.*

Perception du relief

Au terme de ce chapitre, il apparait donc assez clairement que c'est surtout notre cerveau qui « voit », notre œil ne faisant que collecter des informations. Une dernière preuve peut être apportée par nos rêves. Nous voyons des scènes bien *réelles*, générées par notre cerveau, alors que nos yeux sont clos hermétiquement et l'espace qui nous entoure bien noir.

Il est temps maintenant, après avoir assimilé le rôle du cerveau dans le domaine de la vision, d'approfondir les effets de la perception en trois dimensions.

CHAPITRE 2

LA TROISIEME DIMENSION

Jusqu'à présent nous ne nous sommes pas occupés de la qualité de restitution des images perçues par nos yeux et interprétées par le cerveau. La réception d'une certaine quantité d'énergie lumineuse était traduite en impulsions électriques, en concordance avec les paramètres définissant cette entité.

Cependant la vision humaine est capable d'atteindre un haut niveau de performances, lors de l'analyse visuelle des objets composant notre environnement. La survie et le développement intellectuel de l'espèce humaine passent nécessairement par des possibilités dépassant largement le fait de ne percevoir que de vagues *taches* lumineuses.

Le cerveau est un outil d'analyse et de synthèse merveilleusement performant, aussi bien en qualité de restitution qu'en temps de réponse. Quel ordinateur serait aujourd'hui capable de percevoir des formes avec un pouvoir de *résolution* aussi fin et une telle rapidité d'interprétation (le véritable *temps réel)*?

Même les effets de la distance sont rendus de façon à nous permettre d'appréhender celle-ci avec une précision extraordinaire. Une simple flèche dirigée par l'homme vers son but en est l'exemple typique.

Enfin et surtout, nous pouvons nous déplacer autour d'un objet tout en ayant une restitution parfaitement cohérente, non seulement de cet objet, mais aussi de tout l'espace qui l'intègre.

1. LA PERCEPTION DES FORMES

De récentes découvertes scientifiques ont permis de cerner avec davantage de précision les mécanismes complexes du cerveau qui concourent à la perception des formes.

Perception des formes

Examinons par exemple le cas, pour simplifier, où nous observons un simple vase posé sur une table. Le cerveau n'analyse pas instantanément à un endroit de lui-même localisé avec précision, les impulsions électriques reçues, dans le but d'en dresser une image fidèle et complète qui nous permettra de dire: « c'est un vase ».

En réalité il va procéder par analyses successives de plus en plus approfondies, en commençant par faire travailler ses zones superficielles, pour terminer par celles situées de façon la plus interne. Ces couches superposées ont toutes une fonction bien précise et c'est par leurs apports d'informations conjugués que peu-à-peu le vase prendra forme, couleur et sera reconnu comme tel.

Dans un premier temps, la couche la plus externe analysera simplement le jeu d'ombres et de lumières, afin de localiser des emplacements bien définis de zones lumineuses. Puis la couche suivante prendra le relais de l'information, afin de détecter avec plus de précision la frontière de différenciation de ces zones.

Et ainsi de suite, de couches en couches de plus en plus profondes, le dessin du vase va se constituer petit à petit. A un moment donné son contour sera clairement défini. Puis sa teinte générale. Ensuite les détails de plus en plus précis qui le composent et leurs différentes couleurs. Jusqu'au moment où l'analyse sera suffisamment exhaustive ou que le pouvoir séparateur lié à l'œil ne permettra plus d'affiner davantage la recherche.

Analyse de la perception

A ce moment la synthèse de cette recherche en chaîne est véhiculée dans une autre partie du cerveau qui, par de nombreuses comparaisons avec des formes qu'elle a appris à connaître, tentera de classer cet objet avec le plus de précision possible.

Finalement le résultat de ce classement transitera à son tour vers le centre de la parole, pour lui associer, selon les mêmes principes de classements, un mot connu.

Nous sommes alors capables , après avoir **vu** notre objet, de lui donner le nom de « VASE ». Soit de le prononcer, soit de l'écrire, ou tout simplement en le mentionnant mentalement, ces interprétations faisant à leur tour intervenir des traitements de l'information complexes. Tout ceci n'a duré cependant qu'un temps inmesurable.

Il suffit simplement de songer à ce qui se passe lors d'un assaut d'escrime ou d'une course automobile à 300 Km/h, pour juger des vitesses phénoménales des différentes analyses, synthèses, interprétations et réactions diverses, ainsi que de la puissance des moyens mis en oeuvre.

2. LES EFFETS DE LA DISTANCE

Continuons à nous préoccuper de l'extraordinaire puissance d'analyse du cerveau. A présent il va devoir résoudre un cas épineux: « si tel objet est vu comme étant plus petit que tel autre, est-ce réellement le cas, ou est-ce simplement un effet trompeur dû à la distance qui les sépare ? ».

En effet, chacun de nous a remarqué dès sa plus tendre enfance que les objets éloignés apparaissent petits et les proches grands. Que lorsque nous avançons, les objets situés dans un plan rapproché semblent se déplacer plus rapidement que ceux relativement plus éloignés.

Pourtant les enfants éprouvent de la difficulté à transcrire sur le papier ce qu'ils ressentent: la route qu'ils dessinent possède la même largeur à ses deux extrémités, ce qui fait que la traditionnelle maison paraît perchée au sommet d'un piquet, plutôt que située au bord d'un chemin. Ceci parce qu'ils **savent** que la route est de même largeur tout au long de son tracé et que leur cerveau ne possède pas encore l'expérience nécessaire à l'interprétation écrite d'informations visuelles trompeuses. En bas âge ils ont même véritablement l'impression que les objets se déplacent réellement lorsqu'ils se meuvent, en les regardant.

Angle de vision

En outre, un objet situé près de nous semble se rétrécir vers l'horizon. La locomotive d'un train venant à notre rencontre possède une dimension bien définie, mais ses derniers wagons paraissent ne plus former qu'une ligne. Tout est fonction d'un *angle de vision*, entre les extrémités d'un objet et l'œil: plus l'objet est éloigné, plus l'angle diminue. Réciproquement cet angle augmente avec un éventuel rapprochement. (La figure 2.1 en illustre le principe). C'est ainsi qu'un ballon tenu à bout de bras est capable de cacher l'immensité de la sphère solaire.

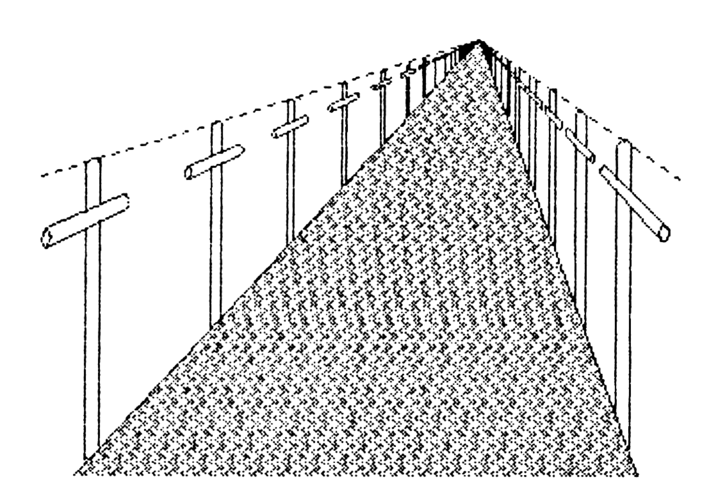


Figure 2.1 — Un des effets de la perspective: les objets paraissent rapetissés, en fonction de leur éloignement de l'observateur.

C'est encore le cerveau, qui, par des prouesses de déductions logiques et par expérience, est capable de se rendre compte que des objets connus sont situés à des distances différentes. Avec les formes il avait aussi mémorisé toutes les caractéristiques de celles-ci, y compris les effets dûs à leurs tailles respectives.

Sachant qu'un arbre est plus petit qu'un crayon, si ce dernier semble plus grand que le premier, c'est qu'il ne peut en être que plus rapproché. Mais ce ne sont pas les seules caractéristiques permettant de fixer la distance d'un objet. Nous verrons plus loin que la perception du **relief** tient un rôle prépondérant au niveau de ces interprétations.

Perspective

Cependant, les effets de la distance peuvent être suggérés dans un tableau de maître, alors que la représentation n'est effectuée que suivant deux dimensions. Les effets du relief et des différences de taille des formes connues se trouvent augmentés par le fait que nous percevons les détails des objets rapprochés avec beaucoup plus de netteté que ceux éloignés. Les couleurs sont de même plus vives et mieux définies pour des formes proches. Ainsi un paysage montre toujours l'éloignement des montagnes par des teintes grises et des contours un peu flous.

3. LES EFFETS DU POINT DE VUE

Point de vue

Toujours en poursuivant l'investigation de nos recherches concernant la vision d'objets dans l'espace, il est important de souligner les effets du *point de vue* sur la représentation de ces images. Nous nommerons *point de vue* la position *ponctuelle* qu'occuperait un œil, par rapport à l'objet observé. Ceci pour simplifier, car il s'agit en fait de la situation spatiale de l'observateur.

Les effets de ce point de vue se manifestent de deux façons différentes: un effet de perspective et l'occultation des surfaces cachées. En ce qui concerne la perspective, la figure 2.1, représentant une route bordée de poteaux télégraphiques, montre bien la sensation de *fuite* des lignes horizontales vers un point précis, situé à l'horizon. L'espacement entre chaque poteau paraît de plus en plus rétréci, en fonction de l'éloignement.

La figure 2.2 fait apparaître, en plus, les effets de la distance entre le point de vue et l'objet : plus l'observateur est proche de l'objet (A), plus les effets de la perspective se font sentir et plus les lignes droites périphériques présentent un rayon de courbure prononcé. Par contre, avec un éloignement beaucoup plus important (B), ces mêmes lignes restent rectilignes et les effets de la perspective sont beaucoup moins marqués.

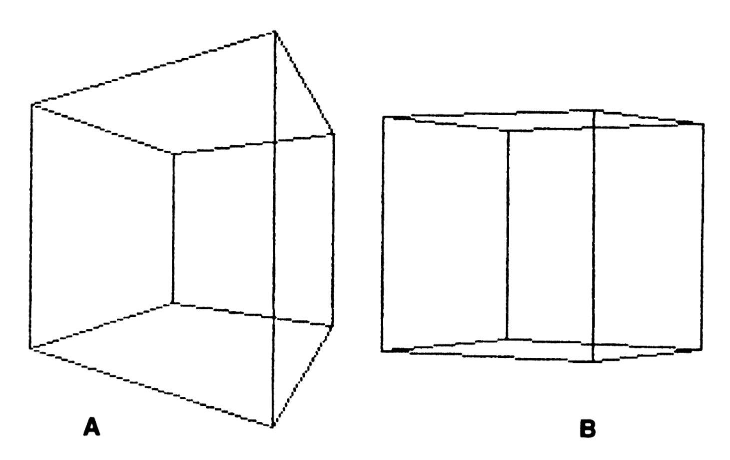


Figure 2.2 — Un autre effet de la perspective :plus l'observateur s'approche d'un objet, plus les déformations apparentes sont accentuées. En « A », le cube est proche de l'observateur. Il en est beaucoup plus éloigné en « B ».

Focales

Les personnes pratiquant la photographie ont pu se rendre compte de ce principe, par l'utilisation d'objectifs *grands angulaires* et *longues focales*: un monument ou un portrait, photographiés à l'aide d'un *grand angle*, montrent des déformations plus ou moins prononcées. A moins de désirer obtenir des effets spéciaux, il est préférable, pour ces prises de vues, d'utiliser une longue focale. En simulant les effets d'un éloignement relatif, les proportions seront ramenées à des valeurs plus esthétiques.

Il est donc important, lorsque l'on précise la position d'un *point de vue*, de définir non seulement sa situation exacte par rapport à l'objet observé, mais aussi la distance de l'objet à laquelle il se trouve.

Surfaces cachées

La figure 2.3 reproduit les effets des surfaces cachées. Les zones d'un objet, situées derrière d'autres parties opaques par rapport au point de vue, seront évidemment invisibles. En cas de déplacement de l'observateur autour de cet objet, les parties cachées seront différentes.

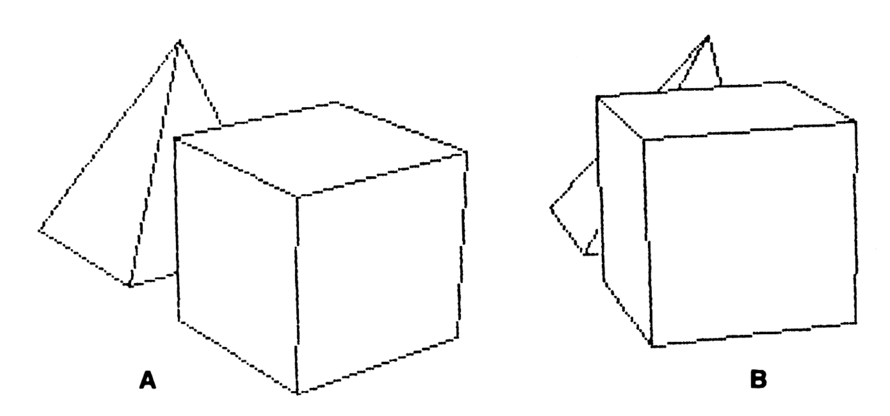


Figure 2.3 — Modification de la vision des objets cachés, en fonction de la situation spatiale de l'observateur. En « A », l'observateur est situé largement à gauche du cube, tandis qu'en « B il s'est déplacé vers la droite.

Situation spatiale

Le cerveau doit tenir compte de toutes ces considérations afin de maximiser l'information permettant de nous situer avec précision au sein de notre environnement. Il apparaît donc parfaitement évident que les messages visuels sont largement interprétés, dans le but d'en extraire le maximum d'enseignements.

Pourtant la nature a prévu de nous doter de deux yeux. Ce n'est pas uniquement en vue de doubler les organes réceptifs par souci de sécurité, mais aussi pour permettre à l'homme d'appréhender l'espace avec davantage de précision. C'est ainsi que nous avons la faculté de voir en **RELIEF**.

CHAPITRE 3

LA PERCEPTION DU RELIEF

La vision en relief est un don de la nature, qui permet aussi bien aux animaux qu'à l'homme de posséder une idée « assez exacte » de l'environnement. Lorsque nous fermons un œil, nous n'avons qu'une impression *approchée* de la distance séparant les différents plans d'un paysage, par exemple.

Par contre, avec nos deux yeux, nous *voyons* réellement cette distance. C'est ce qui permet à un singe de sauter rapidement de branche en branche, ou à un conducteur automobile de négocier un virage instinctivement. La différence de sensation est phénoménale et jusqu'à présent il était vraiment dommage d'en priver nos chers ordinateurs.

Etudions donc avec un peu plus d'attention les mécanismes concourant à la vision en *relief* à l'aide de nos deux yeux d'une part, mais surtout du pouvoir de synthèse de notre cerveau d'autre part.

1. POURQUOI DEUX YEUX ?

Nous l'avons vu précédemment, il est réellement très utile de posséder deux yeux. En cas d'accident, la perte d'un œil n'est pas catastrophique, nous pouvons continuer de voir avec l'autre. Seulement notre vision ne sera plus aussi performante, car la notion de relief aura disparu, du moins dans sa forme la plus importante.

Relief et champ visuel

De récentes découvertes ont permis d'approfondir la connaissance au niveau de la perception du relief. Contrairement à ce que l'on pensait jusqu'à présent, chaque œil possède un rôle déterminant dans ce type de vision. La réunion de leurs informations complète exhaustivement la qualité du message transmis au cerveau.

Comme le montre la figure 3.1, le *champ visuel* se décompose principalement en deux zones : le champ visuel gauche et le champ visuel droit.

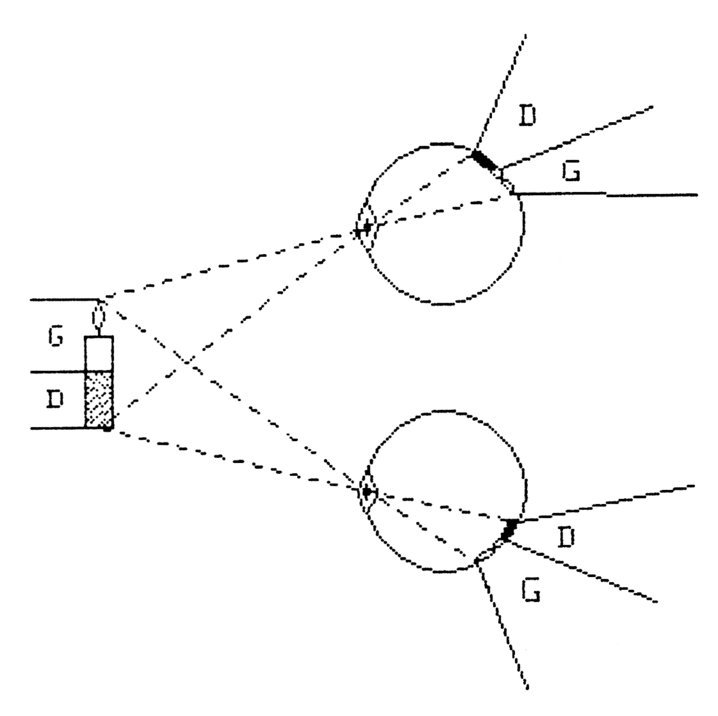


Figure 3.1 — Décomposition du champ visuel en deux parties, droite et gauche, à l'intérieur de chaque œil. L'écart des deux yeux étant sensible, des différences d'interprétation apparaîtront au niveau du cerveau.

L'image étant inversée sur la rétine, la partie droite d'un œil correspond aux informations du champ visuel gauche et vice versa. Ceci étant, les fibres optiques partant de chaque œil n'aboutissent pas toutes dans la même zone du cerveau. Celles issues de la partie droite d'un œil se dirigent dans l'hémisphère droit du cerveau, tandis que celles en provenance de la partie gauche concernent l'hémisphère gauche. (figure 3.2)

Chiasma optique et inversion visuelle

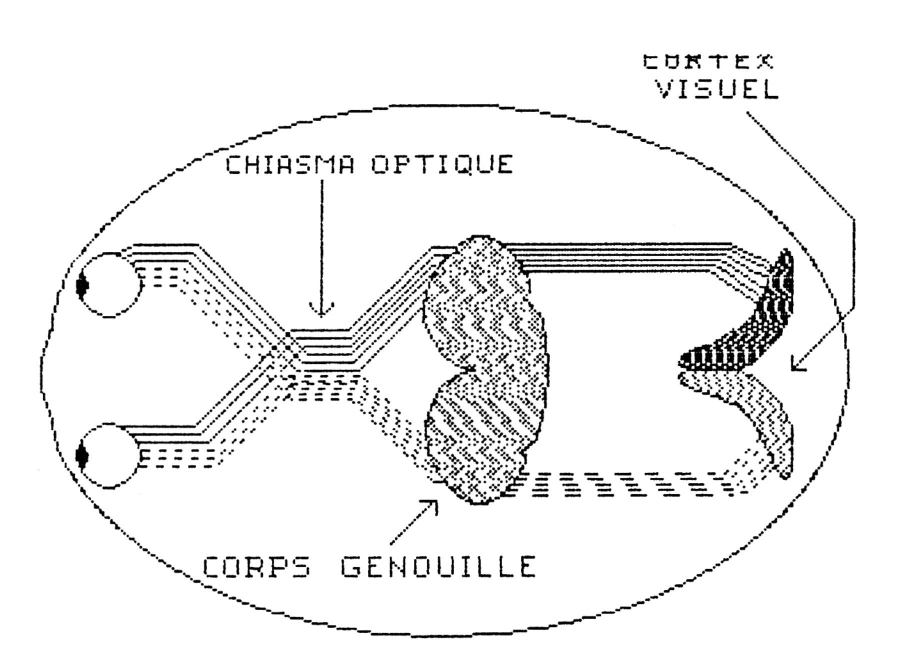


Figure 3.2 — Croisement des informations issues des yeux, au niveau du chiasma optique. Les champs visuels de même nature sont regroupés, afin que leurs différences soient analysées, dans le corps genouillé et le cortex visuel.

Ce principe suffit à expliquer les mouvements simultanés et synchronisés des deux yeux, afin que les observations soient cohérentes.

2. LE CERVEAU, UN SYNTHETISEUR

Synthèse des informations

Il y a donc un croisement au niveau des fibres des nerfs optiques, dénommé *chiasma optique*. De la sorte, chaque hémisphère est alimenté par les informations des deux yeux, lesquelles concernent soit le champ optique droit, soit le gauche. Le résultat de tous ces croisements fait que la partie gauche d'un objet observé est analysée par la partie droite du *cortex* visuel et inversement.

En fait, le traitement de l'information est beaucoup plus complexe, car les *corps genouillés* participent aussi à cette synthèse visuelle, ainsi que les zones avoisinantes et plus profondes du cerveau.

Relief et angle de vision

Du fait de l'écartement des yeux (environ 7 centimètres), l'image observée par un œil est légèrement différente de celle observée par l'autre. Ces différences sont d'autant plus sensibles que la distance de l'objet aux yeux diminue. Effectivement, l'angle de vision (« A », figure 3.3) compris entre les deux côtés du triangle formé par l'objet d'une part et les yeux d'autre part, varie en fonction de la distance du point observé. Plus l'objet est proche, plus cet angle sera grand. Ainsi les différences des observations effectuées par chaque œil seront d'autant plus sensibles Le cerveau sera alors capable d'en déduire, avec une exactitude satisfaisante, la distance à laquelle est situé l'objet.

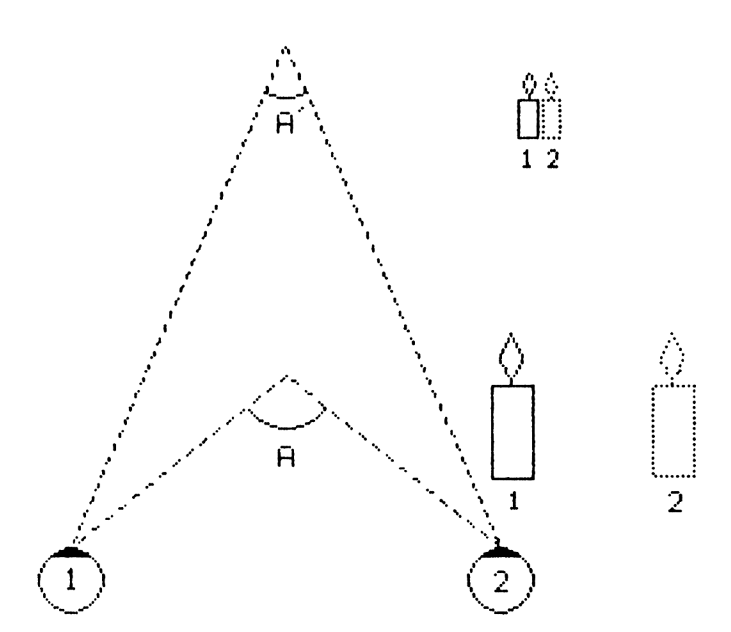


Figure 3.3 — L'importance de l'angle de vision (A et A'), en fonction de l'éloignement des objets. Le cerveau a besoin de cette information, afin d'en déduire la distance à laquelle se situe l'objet.

Image visuelle

Par une synthèse de toutes les informations mémorisées dans d'autres zones du cerveau, l'image visuelle du départ est enfin transformée en une image cérébrale. C'est ainsi que ce message nous est transmis avec la sensation de « voir » l'espace séparant plusieurs objets, échelonnés dans divers plans spaciaux.

Image cérébrale

C'est ce que nous nommons le **RELIEF**.

A présent, nous disposons de toutes les données qui permettent d'expliquer la vision en relief. Il reste à définir les principes « physiques », capables de générer des informations relatives à une sensation de relief par le cerveau, alors qu'en réalité la représentation n'en sera que bidimensionnelle.

MATERIEL	GRILLE	MODE H.R.	EFF. ECRAN	COULEUR	TRACE 1 POINT	TRACE 1 DROITE
TRS 80 HRC	320×250	implicite	CLS, HCLS	HCOLOR c	HPLOT x1,y1	HPLOT @ TO x2,y2
APPLE	280×192	HGR2, HGR	HOME (écran texte) HGR, HGR2	HCOLOR=c (0 à 7)	HPLOT x1,y1	HPLOT x1,y1 TO x2,y2
SPECTRUM	176×256	-------	-------	-------	PLOT x1,y1	DRAW x2,y2
ATMOS	240×200	HIRES	-------	INK c	CURSET x1,y1,c	DRAW x2,y2,c
THOMSON	320×200	-------	-------	-------	PSET (x1,y1),c	LINE (x1,y1)−(x2,y2),c
COMMODORE 64	320×200	GRAPHIC	PRINT (TE)	PRINT (TC)	PLOT (x1,y1,1)	(DRAW x2,y2,1)
NEWBRAIN	640×250	OPEN,11,W640	-------	-------	DOT (x1,y1,c)	DRAW (x2,y,c)
SPECTRA VIDEO	-------	SCREEN 1,1	-------	-------	PSET (x1,y1),c	LINE (x1,y)−(x2,y2),c
HEWLETT PACKARD	-------	GRAPH	GCLEAR	PEN	MOVE x1,y1	DRAW x2,y2
IBM PC	640×200	SCREEN 1 ou 2	CLS	COLOR c	PSET (x1,y1)	LINE (x1,y1)−(x2,y2)
ATARI	-------	GRAPHICS n	GRAPHICS	COLOR c	PLOT x1,y1	DRAW TO x2,y2
MSX	256×192	SCREEN 2	COLOR ce,cf	COLOR ce,cf	PSET (x1,y1),c	LINE −(x2,y2),c
AMSTRAD	640×200	-------	-------	PEN c	PLOT	DRAW

Les seules commandes spécifiques aux différentes machines sont décrites ci-dessus. En effet, seuls **deux** ordres sont nécessaires au tracé des dessins :
– **HPLOT x1,y1** (trace un point, de coordonnées *x1* et *y1*).
– **HPLOT @ TO x2,y2** (trace une droite à partir du dernier point tracé, jusqu'à *x2,y2*).
Car tout dessin peut se résumer au tracé de petits segments de droites, afin de rendre les programmes le plus portables possible.

CHAPITRE 4

COMMENT SIMULER LE RELIEF

Au cours des précédents chapitres, nous avons eu l'occasion d'aborder l'extraordinaire puissance du cerveau, ce dernier étant capable de synthétiser un objet en relief à partir des images formées sur les surfaces de nos rétines. Par des réactions chimiques complexes, il lui est même possible de nous représenter un monde en couleur, au sein d'un univers où tout est irrémédiablement noir...

Nous allons donc profiter de ce pouvoir, afin de lui faire percevoir des formes en relief, à partir de dessins tracés, de même, en deux dimensions.

Pour parvenir à ce résultat, il est nécessaire d'étudier en premier lieu les mécanismes du dessin en trois dimensions, par projection sur une surface plane, puis d'aborder les effets de la situation spatiale de l'observateur par rapport à l'objet. Ensuite, nous résoudrons les problèmes relatifs aux surfaces cachées de l'objet et, enfin, nous forcerons le cerveau à une vision *stéréoscopique* du tracé.

1. LES DESSINS EN TROIS DIMENSIONS

Les périphériques de sortie d'un ordinateur, qu'il s'agisse d'un écran cathodique, d'une imprimante ou d'une table traçante (*plotter*), sont tous des organes ne pouvant représenter que des tracés en deux dimensions. Heureusement, les mathématiques ont depuis longtemps permis la mise en équations de notre environnement tridimensionnel.

De la même façon qu'un artiste est capable de créer une impression de profondeur sur un tableau, à partir d'une scène réelle, il est possible d'afficher ou de dessiner des images en trois dimensions. L'impression de relief sera ainsi suggérée dans un premier temps. Plus loin nous étudierons les techniques de la vue en *vrai relief.* Cependant les principes étudiés pour la représentation en trois dimensions resteront non seulement valables, mais nécessaires.

Pour bien comprendre ces concepts, des bases mathématiques sont nécessaires. Elles resteront cependant à un niveau élémentaire. Voici tout d'abord les différents systèmes de coordonnées.

LES SYSTEMES DE COORDONNEES

Coordonnées bidimensionnelles

Avant d'aborder les systèmes de coordonnées tridimensionnels, il est vivement conseillé de bien assimiler les principes concernant ceux à deux dimensions. L'extension à la troisième dimension s'effectuera alors beaucoup plus aisément.

L' espace à deux dimensions

Coordonnées cartésiennes

Un dessin sur papier ou sur écran, donc compris dans un espace à deux dimensions, peut être représenté sous forme de coordonnées **cartésiennes ou polaires**. Nous conserverons les traditionnelles représentations mathématiques, afin de ne pas dérouter le lecteur.

Coordonnées polaires

Ainsi, en deux dimensions, le système de coordonnées est représenté par l'intersection de deux axes, **X,X'** et **Y,Y'** (figure 4.1). En représentation *cartésienne*, les coordonnées du point « **P** » seront notées par rapport à « **O** » **XP,YP** alors qu'en système *polaire* l'angle **THETA** (θ) ainsi que la distance **RHO** (ρ) suffisent à déterminer parfaitement la situation de ce point sur le plan.

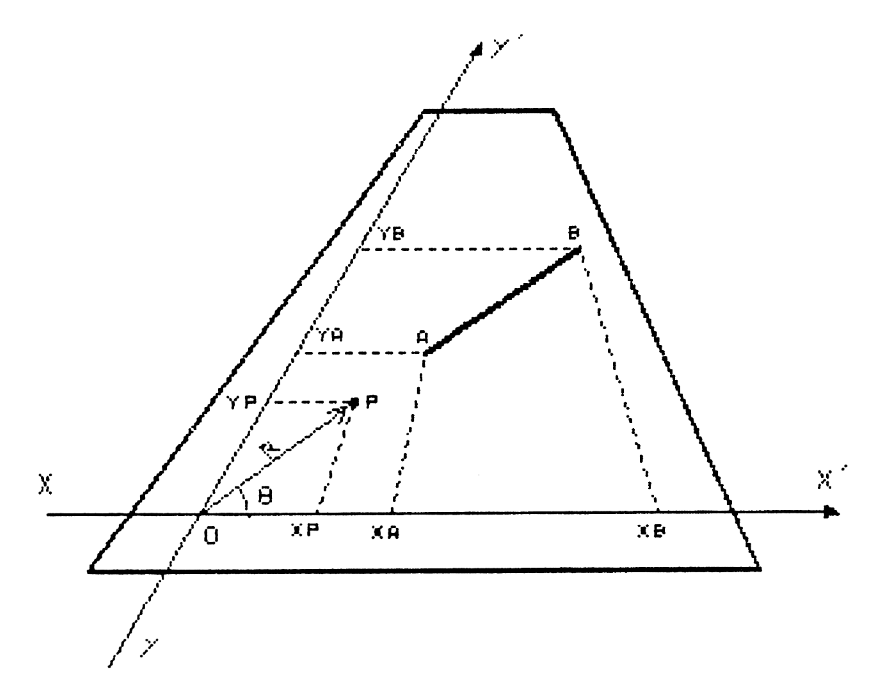


Figure 4.1 — Représentation du système de coordonnées bidimensionnel, matérialisé par l'intersection de deux axes (XX' et YY').

Pour passer facilement d'un système à l'autre, il suffit d'appliquer les formules élaborées par EULER (1707-1783):

$$x = \varrho \cos \theta \qquad y = \varrho \sin \theta$$

$$\varrho = \sqrt{x^2 + y^2} \qquad \cos \theta = \frac{x}{\varrho} \qquad \sin \theta = \frac{y}{\varrho} \qquad \text{(pour } \varrho \text{ différent de 0)}$$

De ces deux systèmes, le plus connu est celui immortalisé par **René DESCARTES** (1596-1650), bien que sa découverte remonte à l'antiquité. Sur un écran de visualisation, les coordonnées *cartésiennes* sont aussi les plus faciles à représenter, par construction. En effet, la figure 4.2 représente un écran dont l'accès aux différents *pixels* (les points pouvant être allumés ou éteints) est calqué sur les coordonnées cartésiennes:

Afin d'allumer le point « **P** » ayant pour coordonnées XP= 50, YP=40, il suffit de programmer, par exemple, la ligne suivante:

HPLOT 50,40

Et pour tracer le *vecteur* (ou segment de droite) **AB** dont les coordonnées de **A** sont XA= 60, YA= 70 et celles de **B**, XB= 250, YB=125:

HPLOT 60,70 TO 250,125

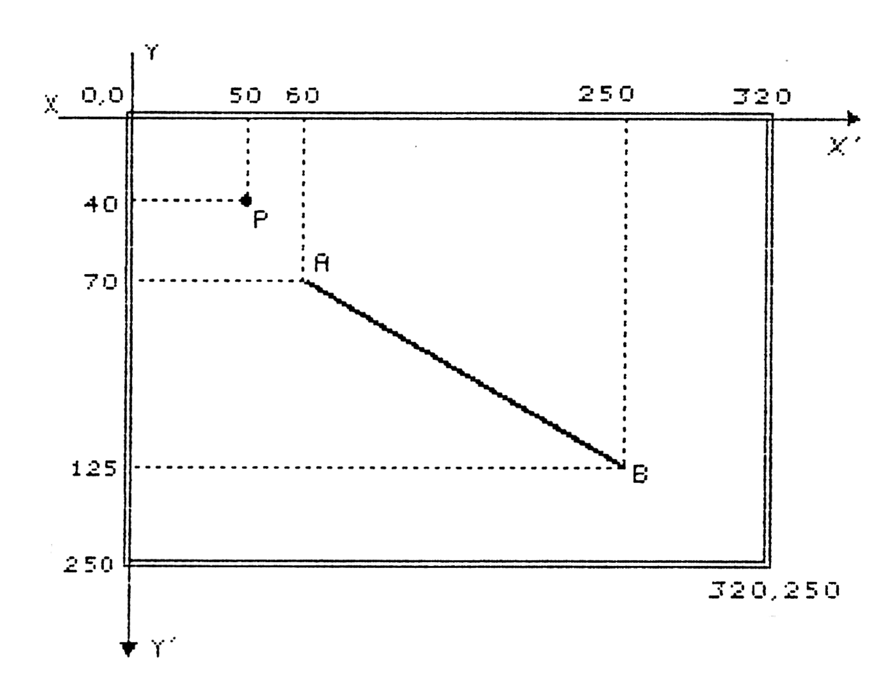


Figure 4.2 — Accès aux différents «pixels» d'un écran de visualisation, selon le système de coordonnées catésiennes.

Remarquez, en outre, que sur un écran l'axe des y est généralement inversé par rapport à celui de la représentation mathématique conventionnelle. Le principe reste bien évidemment identique.

L'espace à trois dimensions

Coordonnées tridimensionnelles

Les concepts énoncés ci-avant restent valables pour la représentation d'objets dans un espace tridimensionnel, mis à part le fait que maintenant un troisième axe sera à considérer, perpendiculaire à ceux déjà cités, l'axe **Z,Z'** (figure 4.3).

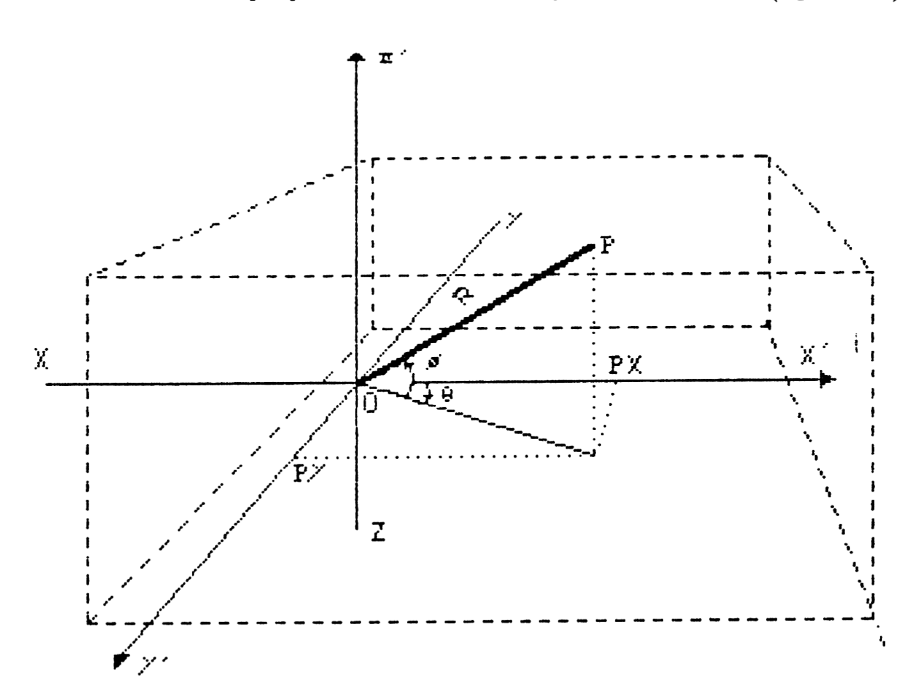


Figure 4.3 — Représentation du système de coordonnées tridimensionnel, matérialisé par l'intersection de trois axes (XX', YY' et ZZ').

Ici aussi les coordonnées peuvent s'inspirer soit du système *cartésien*, soit du système *sphérique*, avec les particularités suivantes :

Coordonnées cartésiennes ou sphériques

En représentation *cartésienne*, les coordonnées du point « **P** » seront relevées par rapport à « **O** » comme étant égales à PX, PY et PZ. En représentation *sphérique*, il suffira de définir *RHO* (ρ), l'azimut *THETA* (θ) et le site *PHI* (φ).

De même, pour passer aisément d'un système à l'autre, les formules suivantes seront à appliquer :

$$x = \varrho \cos \theta \cos \varphi$$
$$y = \varrho \sin \theta \cos \varphi$$
$$z = \varrho \sin \varphi$$
$$\varrho^2 = x^2 + y^2 + z^2$$

Or les systèmes de visualisation, sur micro-ordinateur, ne comportent toujours que les deux axes de coordonnées, **X** et **Y**. Le principe du tracé bidimensionnel reste donc applicable, en ce qui concerne ces axes. Le troisième axe, Z, sera imaginaire et s'éloignera de l'observateur.

Ce sont des transformations mathématiques, basées sur le *calcul matriciel*, qui vont permettre entre autre l'application de la troisième dimension au niveau d'une surface plane.

QUELQUES RAPPELS SUR LES MATRICES

Les matrices sont habituellement représentées sous forme de tableaux :

Matrices

$$M = \begin{vmatrix} 2 & 0 \\ 4 & -5 \end{vmatrix} \qquad N = \begin{vmatrix} 1 & 2 \\ 0 & 3 \end{vmatrix}$$

Pour additionner deux matrices il suffit de faire la somme de leurs membres, en correspondance :

$$P = M + N = \begin{vmatrix} 2 & 0 \\ 4 & -5 \end{vmatrix} + \begin{vmatrix} 1 & 2 \\ 0 & 3 \end{vmatrix} = \begin{vmatrix} (2)+(1) & (0)+(2) \\ (4)+(0) & (-5)+(3) \end{vmatrix} = \begin{vmatrix} 3 & 2 \\ 4 & -2 \end{vmatrix}$$

Pour les multiplier, la règle **LICOL** sera utilisée, c'est-à-dire qu'une **LI**gne de la première matrice sera multipliée par une **COL**onne de la deuxième :

Règle LICOL

$$Q = M \times N = \begin{vmatrix} 2 & 0 \\ 4 & -5 \end{vmatrix} \times \begin{vmatrix} 1 & 2 \\ 0 & 3 \end{vmatrix} = \begin{vmatrix} (2)(1)+(0)(0) & (2)(2)+(0)(3) \\ (4)(1)+(-5)(0) & (4)(2)+(-5)(3) \end{vmatrix} = \begin{vmatrix} 2 & 4 \\ 4 & -7 \end{vmatrix}$$

Ces calculs seront très utiles, lors d'opérations dans les différents systèmes de coordonnées.

Système bidimensionnel

Toutes sortes de transformations sont possibles, à condition de considérer les coordonnées **x** et **y** d'un point comme une matrice :

$$\begin{vmatrix} X & Y \end{vmatrix}$$

En multipliant cette matrice, par une autre de la forme :

$$\begin{vmatrix} M & N \\ P & Q \end{vmatrix}$$

le résultat obtenu sera le suivant :

$$\begin{vmatrix} X & Y \end{vmatrix} \times \begin{vmatrix} M & N \\ P & Q \end{vmatrix} = \begin{vmatrix} (X)(M)+(Y)(P) & (X)(N)+(Y)(Q) \end{vmatrix}$$

$$= \begin{vmatrix} MX+PY & NX+QY \end{vmatrix}$$

$$= \begin{vmatrix} X' & Y' \end{vmatrix}$$

Modifications bidimensionnelles

Les modifications qu'il est possible d'effectuer seront les suivantes :

– AUCUN CHANGEMENT

$$\begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}$$

car $\begin{vmatrix} X & Y \end{vmatrix} \times \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} = \begin{vmatrix} X\cdot 1+Y\cdot 0 & X\cdot 0+Y\cdot 1 \end{vmatrix} = \begin{vmatrix} X & Y \end{vmatrix}$

– CHANGEMENT D'ECHELLE

- multiplication par 4 sur l'axe des X :

$$\begin{vmatrix} 4 & 0 \\ 0 & 1 \end{vmatrix}$$

car $\begin{vmatrix} X & Y \end{vmatrix} \times \begin{vmatrix} 4 & 0 \\ 0 & 1 \end{vmatrix} = \begin{vmatrix} X\cdot 4+Y\cdot 0 & X\cdot 0+Y\cdot 1 \end{vmatrix} = \begin{vmatrix} X\cdot 4 & Y \end{vmatrix}$

- multiplication par 3 sur les 2 axes :

$$\begin{vmatrix} 3 & 0 \\ 0 & 3 \end{vmatrix}$$

– SYMETRIES

- symétrie par rapport à Y :

$$\begin{vmatrix} -1 & 0 \\ 0 & 1 \end{vmatrix}$$

- symétrie par rapport à X :

$$\begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix}$$

- symétrie totale :

$$\begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$$

– CISAILLEMENT

$$\begin{vmatrix} 1 & M \\ N & 1 \end{vmatrix}$$

M déplace vers l'axe des Y.
N déplace vers l'axe des X.

– ROTATION D'UN ANGLE θ DANS LE SENS HORAIRE

$$\begin{vmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{vmatrix}$$

θ étant en radians :

radians = degrés × π/180 degrés = radians × 180/π π = 3,141592654

– TRANSLATION :

Déplacements

Les opérations ci-dessus effectuaient des transformations par rapport à l'origine des axes. Si nous désirons maintenant les effectuer par rapport à un autre point (x'=x+M, y'=y+N), les matrices 2×2 ne sont plus applicables. Il faut introduire une troisième composante, ainsi qu'une matrice 3×3 (permettant l'inversion matricielle). Ceci en vue d'obtenir :

$$\begin{vmatrix} X & Y & 1 \end{vmatrix} \times \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ M & N & 1 \end{vmatrix} = \begin{vmatrix} X+M & Y+N & 1 \end{vmatrix} = \begin{vmatrix} X' & Y' & 1 \end{vmatrix}$$

M effectuant une *translation* par rapport à l'axe **X** et **N** par rapport à l'axe **Y**.

Une *transformation* devient alors :

$$\begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ M & N & 1 \end{vmatrix}$$

dans laquelle **M** et **N** effectuent une translation de l'origine.

Puis une rotation d'un angle θ:

$$\begin{vmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

En résumé, les actions possibles sur A,B,C,D,M,N seront de la forme :

$$\begin{vmatrix} A & B & 0 \\ C & D & 0 \\ M & N & 1 \end{vmatrix}$$

Une application de ces principes est fournie par le programme 4.1.

On réserve un peu de place pour les chaînes.

```
1 '                        PROGRAMME I.4.1.2.1.
2 '
3 '
10 '****************************************************
20 '***     DEMO DE TRANSFORMATIONS MATRICIELLES EN 2D    ***
30 '****************************************************
40 '
50 CLEAR 500
60 '
70 '****************  fonctions utilisees  ****************
80 PI=3.14159
90 DEF FNMX(X,Y)= XX*X + XO*Y    'modif. coord. X par matrice
100 DEF FNMY(X,Y)= YO*X + YY*Y    'modif. coord. Y par matrice
110 DEF FNUN(N)= INT(N*10+.5)/10  'arrondi 1 decimale
120 DEF FNRA(A)= A*PI/180         'ANGLE degres --> radians
130 GOTO 330
140 '
150 '=====================================================
160 '=====================================================
170 '              BLOC DES COMMANDES SPECIFIQUES
180 'Definition ecran Basse Resolution: 16 lignes, 64 colonnes
190  MX=319: MY=249:  RETURN  'Def. surf. ecr. (X max et Y max)
200  ZX=INT(MX/2): ZY=INT(MY/2): RETURN  'Calcul coord. centre
210 HPAGE  0: RETURN 'Selection 1 ere page graphique
220 HCLS   0: RETURN 'Effacement ecr. graphique, couleur noire
230 HDISP  1: RETURN 'Visu Basse Resolution
240 HDISP  2: RETURN 'Visu Haute resolution
250 HCOLOR 2: RETURN 'Couleur du trace (VERT)
260 HCOLOR 7: RETURN 'Couleur du trace (BLANC)
270 HCOLOR 1: RETURN 'Couleur du trace (ROUGE)
280 'HPLOT x,y TO x',y' => trace d'une droite (de x,y a x',y')
290 '=====================================================
300 '=====================================================
310 '
320 'Initialisations
330 GOSUB 190: GOSUB 200: GOSUB 210: GOSUB 220: GOSUB 230: CLS
340 IF Z=1 THEN GOTO 420  'Detect. 1er passage (Z=0 apres RUN)
350 Z=1
360 '
370 'Definition coordonnees carre + matrice de transformation
380 AX=0:  AY=0:           XX=3: YO=0
390 BX=30: BY=0:           XO=0: YY=2
400 CX=30: CY=30
410 DX=0:  DY=30
420 GOSUB 750
430 '
440 'Calcul transformations des coordonnees d'apres la matrice
450 EX= FNMX(AX,AY): FX= FNMX(BX,BY): GX= FNMX(CX,CY): HX= FNMX(DX,DY)
460 EY= FNMY(AX,AY): FY= FNMY(BX,BY): GY= FNMY(CX,CY): HY= FNMY(DX,DY)
470 '
480 'report des coordonnees sur origine des axes xx' et yy'
490 EX=EX+ZX: EY=EY+ZY: FX=FX+ZX: FY=FY+ZY: GX=GX+ZX: GY=GY+ZY: HX=HX+ZX: HY=HY+ZY
500 '
510 'Verification non depassement limites ecran
520 IF (EX>MX) OR (FX>MX) OR (GX>MX) OR (HX>MX) OR (EY>MY) OR (FY>MY) OR (GY>MY) OR (HY>MY) THEN
 GOSUB 230: CLS: PRINT "DEPASSEMENT LIMITES ECRAN": RUN
530 '
540 'Affichage des axes de coordonnees xx'et yy'
550 GOSUB 240: GOSUB 250
560 HPLOT 0,ZY TO MX,ZY
570 HPLOT ZX,0 TO ZX,MY
580 '
590 'Trace du carre d'origine a partir de l'origine des axes
600 GOSUB 250
610 XA=AX+ZX: YA=AY+ZY: XB=BX+ZX: YB=BY+ZY: XC=CX+ZX: YC=CY+ZY: XD=DX+ZX: YD=DY+ZY
620 HPLOT XA,YA TO XB,YB: HPLOT XB,YB TO XC,YC: HPLOT XC,YC TO XD,YD: HPLOT XD,YD TO XA,YA
```

Elles évitent les erreurs de frappe.

Ce bloc est à adapter suivant le type de votre machine.

Le programme débute ici.

Par exemple...

Afin d'éviter l'arrêt du programme.

HPLOT trace une droite ici. A adapter selon le langage.

Programme 4.1 ▶▶▶

▶▶▶

Tapez une touche pour continuer.

Et voici le résultat !

HPLOT trace une droite. A adapter selon le BASIC.

Une saisie d'écran simplifiée.

Conserver les anciens paramètres.

On désire apporter des modifications...

... soit sur l'angle de rotation, soit sur la matrice.

Réaffichage des mises à jour.

```
630 '
640 'Attente avant trace suivant
650 R$=INKEY$: IF R$="" THEN 650
660 '
670 'Trace du carre translate
680 GOSUB 270
690 HPLOT EX,EY TO FX,FY: HPLOT FX,FY TO GX,GY: HPLOT GX,GY TO HX,HY: HPLOT HX,HY TO EX,EY
700 '
710 'Attente frappe d'une touche avant retour basse resolution
720 R$=INKEY$: IF R$="" THEN 720 ELSE GOTO 330
730 '
740 '**************** S/P MODIF PARAMETRES ******************
750 CLS: PRINT "            CARRE                                    MATRICE": PRINT: OX= FNUN(XX):
 PO= FNUN(YO): QY= FNUN(YY): RO= FNUN(XO)
760 PRINT "A:"AX","AY"        B:"BX","BY"                     E:"OX"    F:"PO
770 PRINT
780 PRINT      "D:"CX","CY"      C:"DX","DY"                   G:"RO"    H:"QY
790 PRINT: PRINT STRING$(61,"-"): PRINT
800 PRINT "            Desirez-vous modifier ces donnees (O/N) ?"
810 R$=INKEY$: IF R$="" THEN 810
820 '
830 '******** detection pas de modification demandee *******
840 IF (R$<>"o") AND (R$<>"O") THEN PRINT: PRINT "Appuyer a chaque fois sur une touche pour avan
cer ..."
850 IF (R$<>"o") AND (R$<>"O") THEN S$=INKEY$: IF S$="" THEN GOTO 850 ELSE CLS: GOTO 450
860 '
870 '*********** une modification a ete demandee ***********
880 PRINT "Entrer la LETTRE concernee ( <R> pour rotation ): ";
890 L$=INKEY$: IF L$="" THEN 890 ELSE PRINT L$
900 '
910 '************* detection rotation demandee *************
920 IF (L$<>"r") AND (L$<>"R") THEN 970
930 LINE INPUT "Entrer l'angle de rotation en degres : ";AG$
940 AG= FNRA(VAL(AG$)): XX= COS(AG): YO= SIN(AG): XO= YO*(-1): YY= XX: GOTO 1080
950 '
960 '*********** detection modification matrice ************
970 IF (L$="e") OR (L$="E") OR (L$="f") OR (L$="F") OR (L$="g") OR (L$="G") OR (L$="h") OR (L$="
H") THEN INPUT "Entrer le nombre correspondant : "; I: GOSUB 1110: GOTO 1080
980 '
990 '************* modification carre demandee *************
1000 INPUT "Entrer la coordonnee X : "; I
1010 INPUT "Entrer la coordonnee Y : "; J
1020 IF (L$="a") OR (L$="A") THEN AX=I: AY=J: GOTO 1080
1030 IF (L$="b") OR (L$="B") THEN BX=I: BY=J: GOTO 1080
1040 IF (L$="c") OR (L$="C") THEN CX=I: CY=J: GOTO 1080
1050 IF (L$="d") OR (L$="D") THEN DX=I: DY=J: GOTO 1080
1060 '
1070 '******** retour affichage parametres modifies *********
1080 GOTO 750
1090 '
1100 '************* S/P MODIFICATION MATRICE **************
1110 IF (L$="e") OR (L$="E") THEN XX=I: RETURN
1120 IF (L$="f") OR (L$="F") THEN YO=I: RETURN
1130 IF (L$="g") OR (L$="G") THEN XO=I: RETURN
1140 IF (L$="h") OR (L$="H") THEN YY=I: RETURN
1150 '
1160 '*****************************************************
1170 '***************** FIN DU PROGRAMME ******************
1180 '*****************************************************
```

Fin du programme 4.1

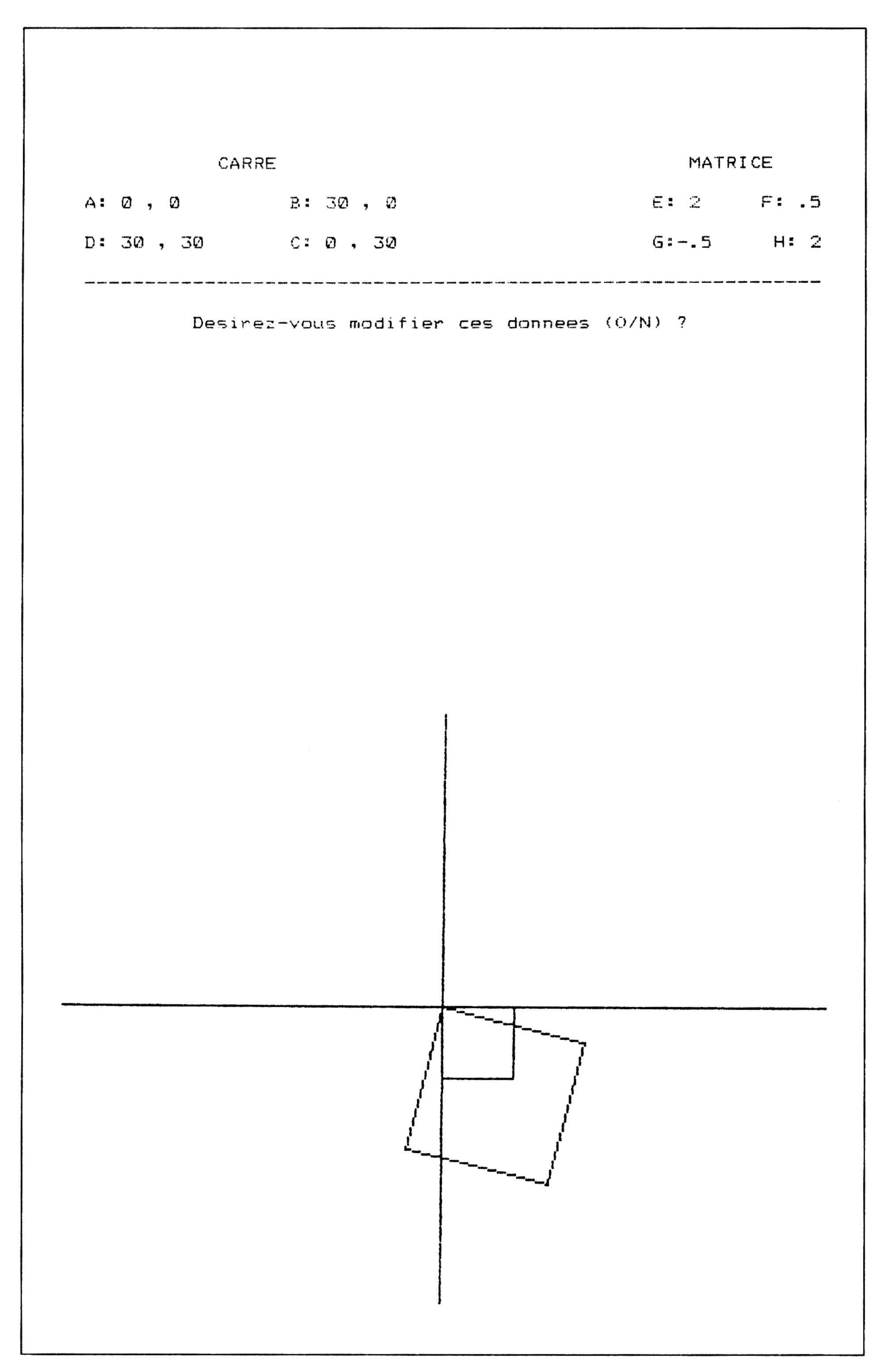


FIGURE 4.4 – Modifications générées par le programme, qui applique une matrice de transformations, à un carré.

Transformations d'un carré

Ce programme permet une démonstration exhaustive des conséquences du calcul matriciel, sur un système de coordonnées. Son objectif est de démontrer l'application de diverses transformations, sur un carré centré par rapport à deux axes. L'exécution débute par l'affichage d'un tableau en basse résolution. A gauche de l'écran, figurent les coordonnées relatives A, B, C et D des sommets du carré. A droite, une matrice carrée E, F, G et H, dont nous allons pouvoir tester les effets sur le carré.

Il est possible de conserver les paramètres entrés par défaut, en répondant par n'importe quelle lettre sauf « O ». Dans ce cas l'affichage passe en haute résolution, trace les axes de coordonnées en blanc sur fond noir, ainsi que le carré originel en vert. L'appui de n'importe quelle touche provoquera l'affichage en rouge du nouveau carré, transformé par la matrice. Un nouvel appui sur une touche quelconque fera retourner au tableau d'affichage et ainsi de suite. Lorsque vous désirerez modifier l'une des données affichées, répondez par « O » à la première question, puis par la lettre dont vous voulez changer les caractéristiques. Entrez alors la nouvelle valeur. Le tableau sera réaffiché avec les données mises à jour.

« R » pour rotation

Une option a été prévue, afin de simplifier la démonstration. Si vous désirez provoquer une rotation du carré, au lieu d'entrer une des huit lettres, tapez « R »; ainsi les différents *sinus* et *cosinus*, applicables à la matrice E, F, G, H, seront calculés automatiquement (et en *radians*), simplement en répondant par la valeur de l'angle de rotation désiré (en *degrés*), à la demande.

Cisaillements

Les conséquences du calcul matriciel sont aisées à appréhender. En modifiant « E », l'action porte sur un déplacement relatif à l'axe des X. « H » agit dans le sens de l'axe des Y. Modifiez « E » et « H » pour agrandir le carré dans les deux sens à la fois. « F » et « G » provoquent des *cisaillements*. Essayez des nombres négatifs, ce qui aura pour conséquence d'effectuer des tracés symétriques par rapport aux axes, avec agrandissements, cisaillements ou non.

Facteur d'agrandissement

Enfin demandez une rotation de 30 degrés, puis de 45 et 90 degrés. Nous constatons que pour 90 degrés, l'effet obtenu est identique à celui des symétries (d'ailleurs les valeurs de la matrice deviennent bien identiques). En remodifiant par la suite « E » et « H », des facteurs d'agrandissement seront introduits. La facilité d'expérimentation permet d'essayer toutes les combinaisons, afin de bien cerner le problème. Essayez de même de modifier les coordonnées A,B,C,D du carré, pour changer sa forme. Les transformations suivent un tracé identique.

BASIC MICROSOFT

Après ces différents exercices, le rôle du calcul matriciel au niveau des transformations de tracés devrait être limpide. Aussi étudions ce programme pour mieux comprendre son fonctionnement. Son affichage est simplifié dans le but d'assurer une compatibilité totale au niveau des différents micro-ordinateurs utilisables. Seul un module est spécifique à chaque appareil. Ceci est inévitable, mais très facile à modifier grâce aux explications fournies et à l'extrême modularité du logiciel. Le reste du programme est classiquement écrit en BASIC Microsoft. Il appartiendra à chacun d'apporter les améliorations éventuelles au niveau de l'affichage et des diverses vérifications nécessaires, ce qui sortirait du cadre de cette étude et détruirait les compatibilités.

Précisons encore que le listage de ce programme a été imprimé directement à partir du logiciel original, sans recopie manuelle. Aucune faute de frappe n'est donc à redouter.

La ligne 50 réserve suffisamment d'espace pour les chaînes de caractères. Sur certains matériels, ce n'est pas indispensable. Afin de simplifier la lecture du texte, des définitions de fonctions ont été programmées aux lignes 90 à 120. Quelques rares BASIC ne permettent pas l'emploi de DEFFN; il est aisé de le simuler par l'apport d'une variable supplémentaire commune, dans un sous-programme.

La pédagogie implique de sacrifier les performances d'exécution au profit de la lisibilité. Ce programme n'est donc pas optimisé. Ceci n'a cependant pas d'influence notable sur la rapidité des calculs, ceux-ci étant peu nombreux. En conséquence, le calcul matriciel a été simplifié par l'utilisation des fonctions **FNMX** et **FNMY** (lignes 90 et 100), permettant de diviser les calculs au niveau des coordonées X, puis Y. Revoyez ce qui a été étudié ci-avant, lors du calcul matriciel.

Tracé d'un trait par « HPLOT »

Les lignes 180 à 280 reprennent les commandes spécifiques à une machine. Ici l'affichage haute résolution permet de dessiner sur un écran de 320 sur 250 points. Il suffit de modifier ces paramètres en conséquence, ainsi que les différentes commandes gérant la haute résolution (HPAGE, HCLS, HDISP et HCOLOR). La fonction de tracé a été simplifiée à l'extrême et utilise la plus répandue : **HPLOT**. En cas de divergence, modifiez en conséquence cette instruction dans le corps du programme.

Les lignes 380 à 410 fixent les valeurs par défaut des sommets du carré et de la matrice, puis le programme exécute le sous-programme d'affichage et de modification des paramètres, s'étendant des lignes 750 à la fin. Les lignes 750 à 810 affichent les deux tableaux, puis en ligne 850 un aiguillage est effectué vers la ligne 450, dans le cas où l'utilisateur ne désire pas modifier ce qui est affiché. Les lignes suivantes détectent classiquement la valeur associée à la lettre qui doit être modifiée. Un cas particulier, toutefois, concerne les lignes 930 à 940. Il s'agit du cas où une rotation a été demandée. L'angle entré en *degrés* est transformé en *radians* par la fonction **FNRA**, puis les composantes E, F, G, H de la matrice sont transformées en *sinus* et *cosinus*, comme indiqué précédemment. Ceci pour éviter d'avoir à effectuer soi-même ces calculs.

L'objet de ce livre concernant essentiellement le domaine de la troisième dimension, la démonstration en deux dimensions a été simplifiée. Les *translations* de coordonnées n'ont pas été prévues, car elles nécessitent l'utilisation de matrices 3×3. Le programme en aurait été alourdi, sans nécessité absolue. A titre d'exercice, le lecteur pourra toujours effectuer lui-même les diverses modifications, selon les méthodes de calcul énoncées jusqu'à présent.

Après ces quelques rappels sur les opérations matricielles et leur utilité dans le domaine du dessin en deux dimensions, leur application au tracé tridimensionnel sera simplifiée.

Système tridimensionnel

Modifications tridimensionnelles

L'étude précédente s'était étendue sur le tracé à deux dimensions, par l'intermédiaire du calcul matriciel. Cette expérimentation était loin d'être inutile, car les dessins simulant la troisième dimension font appel à des concepts assez complexes, mais qui font suite à ce qui vient d'être étudié. Sa parfaite assimilation ne peut être que bénéfique pour la facilité de compréhension de ce qui va suivre.

Les matrices 2×2 ne suffisaient plus aux translations et il a fallu passer aux matrices 3×3 afin de faire référence à un point supplémentaire d'une part et, d'autre part, autoriser la multiplication matricielle. Il parait évident que l'introduction d'une nouvelle coordonnée (**z**) implique à son tour le traitement d'un vecteur d'ordre 4 (x, y, z, 1) afin de permettre dès à présent les translations. Pour les mêmes raisons, la matrice correspondante devra être de la forme 4×4.

– MODIFICATION D'ECHELLE

Echelle

Ce procédé suit le même principe que celui appliqué aux deux dimensions ; l'action portera sur la diagonale nord-ouest / sud-est de la matrice :

$$\begin{vmatrix} X & Y & Z & 1 \end{vmatrix} \times \begin{vmatrix} A & 0 & 0 & 0 \\ 0 & B & 0 & 0 \\ 0 & 0 & C & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} AX & BY & CZ & 1 \end{vmatrix}$$

Toutes les modifications possibles sont obtenues simplement en « jouant » sur A, B et C, égaux ou non.

– ROTATION

Rotation

Au niveau du système bi-dimensionnel, nous avons vu qu'une rotation d'un angle θ était réalisée en appliquant la matrice :

$$\begin{vmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{vmatrix}$$

Ce qui donne les matrices suivantes, selon l'axe de rotation considéré :

- autour de l'axe **Z**:

$$\begin{vmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

- autour de l'axe **X**:

$$\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

- autour de l'axe **Y**:

$$\begin{vmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

– TRANSLATION

Translation

Reprenons les principes similaires bien connus désormais, qui feront qu'un objet tridimensionnel sera déplacé dans l'espace :

$$\begin{vmatrix} X & Y & Z & 1 \end{vmatrix} \times \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ J & K & L & 1 \end{vmatrix} = \begin{vmatrix} X+J & J+K & Z+L & 1 \end{vmatrix}$$

Ce cas est assez simple à mémoriser. Les nouvelles coordonnées sont obtenues à partir de x, y, z en leur additionnant respectivement J, K et L. Ceci, naturellement, pour un seul point. Pour des figures plus complexes, le même raisonnement est applicable, puisque tout tracé complexe peut être ramené à une suite de points ou à des lignes (de longueur variable) tracées entre deux points de coordonnées connues.

– REFLEXION

Réflexion

Ce procédé est actuellement très souvent appliqué lors de la création des superbes images tridimensionnelles calculées automatiquement par ordinateur. Certains objets se reflètent sur des surfaces planes (miroirs, plans d'eau ou murs).

Les points d'un objet et ceux de son reflet sont séparés par un plan de symétrie.

Ainsi, par rapport à un plan **x y**, chaque point verra le signe de sa coordonnée **z** changer de sens. Pour parvenir à ce résultat, il est nécessaire d'appliquer la matrice de transformation suivante:

$$R\ xy = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Afin d'effectuer cette réflexion par rapport aux autres axes, il suffira d'appliquer la matrice appropriée:

$$R\ xz = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R\ yz = \begin{vmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Il est relativement aisé d'effectuer l'une quelconque de ces transformations par rapport à l'origine des axes. Cependant la complexité du procédé augmente singulièrement lorsque les rotations, translations, réflexions ou modifications sont à appliquer par rapport à un point ou à un axe pris arbitrairement dans l'espace. Ce principe nécessite de considérer la notion de « point de vue », dont nous avons déjà entrevu les implications dans le chapitre 2.

2. LA POSITION DE L'ŒIL DANS L'ESPACE

Avant d'aborder avec précision cette notion, quelques concepts supplémentaires sont à assimiler en priorité. Ils serviront à apporter une solution au problème consistant à visualiser un objet automatiquement selon l'angle sous lequel il est observé.

Vecteurs

Nous avons vu plus haut que tout tracé pouvait se subdiviser en une suite de segments de droite, appelés *vecteurs.* Il suffit, par conséquent, de ramener les études de tracés complexes à des problèmes relatifs à un vecteur, pour simplifier. Mais ce n'est pas tout. Pour appliquer une transformation quelconque (donc susceptible d'être assez « torturée ») à toute figure, par rapport à un axe arbitraire, il est possible d'utiliser uniquement la suite des matrices spécifiques dont nous venons de faire la connaissance.

Décompositions

Prenons un exemple. Considérons d'une part une figure occupant un emplacement précis dans l'espace, et d'autre part un segment de droite (parallèle à l'un des axes, X par exemple) devant servir de référence à un effet de rotation de cet objet autour de ce vecteur (P Q). Au lieu de chercher à créer des matrices complexes et spécifiques à chaque cas, il est bien plus simple de décomposer le mouvement en une suite de transformations simples.

Nous allons donc commencer par ramener l'origine P du vecteur à l'origine des trois axes de coordonnées X, Y et Z, c'est-à-dire effectuer une *translation.* A ce moment il sera aisé d'effectuer une *rotation* par rapport à l'axe des X. Enfin une dernière *translation,* de sens inverse à la première, repositionnera la figure à son emplacement, tout en conservant l'effet de rotation.

Ces trois effets sont simples à effectuer individuellement. La transformation finale sera calculée automatiquement en multipliant ces trois matrices entre elles. Le développement complet est donné ci-dessous :

1 - translation :

$$T = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & K & L & 1 \end{vmatrix}$$

Seuls K et L sont à considérer, J concernant l'effet sur X qui n'est pas à prendre en compte au niveau de cet exemple.

2 - rotation :

$$R = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

La rotation choisie est celle relative à l'axe des X.

3 - translation :

$$T' = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -K & -L & 1 \end{vmatrix}$$

Mêmes remarques que lors de la première translation, mais ici K s'est transformé en - K et L en - L, afin d'effectuer la translation en sens inverse de la première.

Transformation finale

Finalement, la matrice de rotation M' sera égale à :

$$M' = T \cdot R \cdot T'$$

Une remarque s'impose. Dans l'exemple que nous venons de considérer, la matrice T' effectuait l'opération inverse de la matrice T. En les appliquant successivement à un point, celui-ci retrouve ses coordonnées d'origine :

$$T \cdot T' = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -K & -L & 1 \end{vmatrix} \times \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & K & L & 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = I$$

Il est donc possible de définir des matrices (T) et leurs inverses (T' ou T^{-1}).

Si l'on désire effectuer une rotation de θ et une rotation de $-\theta$, les matrices correspondantes seront les suivantes :

$$R = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R^{-1} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Ce qui implique bien que $R \cdot R^{-1} = I$

Inversion de matrice

En ce qui concerne la suite de ces développements, trouver l'inverse d'une matrice donnée sera très utile, voire nécessaire. Seulement toutes les matrices ne possèdent pas nécessairement leur inverse. Heureusement, dans le domaine du tracé tridimensionnel, les matrices étudiées précédemment sont toutes inversibles. Il est même assez simple de les calculer, puisque nous avons pu observer qu'il suffisait de changer les montants positifs en leurs homologues négatifs, à l'emplacement voulu (K en −K, L en −L, COS θ en −COS θ, SIN θ en −SIN θ). Ceci pour les translations et les rotations. Au niveau des changements d'échelle, l'inversion s'obtient en transformant P en 1/P.

Tout ceci nous amène progressivement vers la solution idéale, qui consiste simplement à appliquer ces transformations au système d'axe de coordonnées luimême. Cette façon de procéder provoquera des translations et des rotations automatiques d'objets à partir de translations et rotations des axes **X**, **Y** et **Z**, par rapport à l'un d'entre eux, mais en sens inverse. Ainsi pour simuler une rotation θ d'une figure, il suffira de faire subir une rotation −θ au système d'axes.

Effet de « point de vue »

Nous sommes donc parvenus progressivement au problème de la position de l'œil dans l'espace, connu sous le nom d'effet de « point de vue ». En effet, lorsque nous désirons observer un objet, nous pouvons soit le faire tourner autour de lui-même par un moyen quelconque (avec la main ou mécaniquement), soit le laisser immobile. Par contre, dans ce dernier cas, ceci implique que c'est l'observateur qui doit se déplacer tout autour de l'objet observé, afin d'en voir toutes les faces.

Matrice de transformation

C'est ce que nous effectuerons désormais. L'obtention de ce résultat passera par la définition de la matrice de transformation, par rapport à l'objet luimême, en décomposant le mouvement si besoin est. Ensuite des calculs permettront de déduire l'inverse de la matrice finale de transformation. Cette dernière sera alors appliquée au système d'axes de coordonnées. De la sorte tout un ensemble de figures complexes peut être redessiné par une unique modification de la position relative des axes **X**, **Y** et **Z**, dans l'espace. Par conséquent l'application de ce procédé sera considérablement facilitée en notant les coordonnées d'un point relativement à l'origine du système d'axes, au lieu de les considérer en valeurs réelles par rapport à l'écran. Les calculs et l'élaboration des tracés seront beaucoup plus aisés à effectuer.

Un réel effort est nécessaire à la totale compréhension de ce chapitre, afin de parvenir à une assimilation satisfaisante des concepts qui ont été développés ici. Il ne faut pas hésiter à refaire les calculs, pour mieux en comprendre le sens. Un point obscur peut nuire énormément à la saisie de la suite des développements situés à des niveaux supérieurs. Ce ne sera jamais une perte de temps inutile, mais plus précisément un investissement sur l'avenir. A présent il est possible de redescendre au niveau de notre micro-ordinateur, en résolvant de nouveaux problèmes : ceux soulevés par l'utilisation d'une surface plane comme moyen de visualisation (écran cathodique ou feuille de papier imprimée).

3. LA PROJECTION SUR L'ECRAN

Savoir manipuler des figures en trois dimensions ne constitue pas l'aboutissement de nos efforts. Un niveau de la plus haute importance reste à franchir, car nous ne sommes toujours pas capables de traduire ces différents mouvements sur l'écran. Il est en effet nécessaire de traduire des formes évoluant dans la troisième dimension, au sein d'un environnement bidimensionnel. Autrement dit, il faudra effectuer une projection géométrique d'un espace sur l'autre, de niveau inférieur, en reprenant chacun de ses points.

Centre de projection

Un exemple de projection selon les principes énoncés au début de ce chapitre, vous est proposé pour la figure 4.5. Il apparait clairement que pour donner l'illusion de la profondeur, l'image résultante sur le plan devra être déformée en conséquence. Or les projections planes peuvent s'effectuer soit en *parallèle*, soit en *perspective*. Avec ce dernier cas, les lignes fuyantes concourent toutes vers un même point, appelé *centre de projection*, tandis qu'avec le précédent elles ne se rejoignent pas. Le type *parallèle* correspond à la réalité *physique* des objets et est fréquemment utilisé en dessin industriel avec la projection *cavalière* ou *cabinet*. Par contre notre système de vision est tout à fait adapté au type en *perspective*.

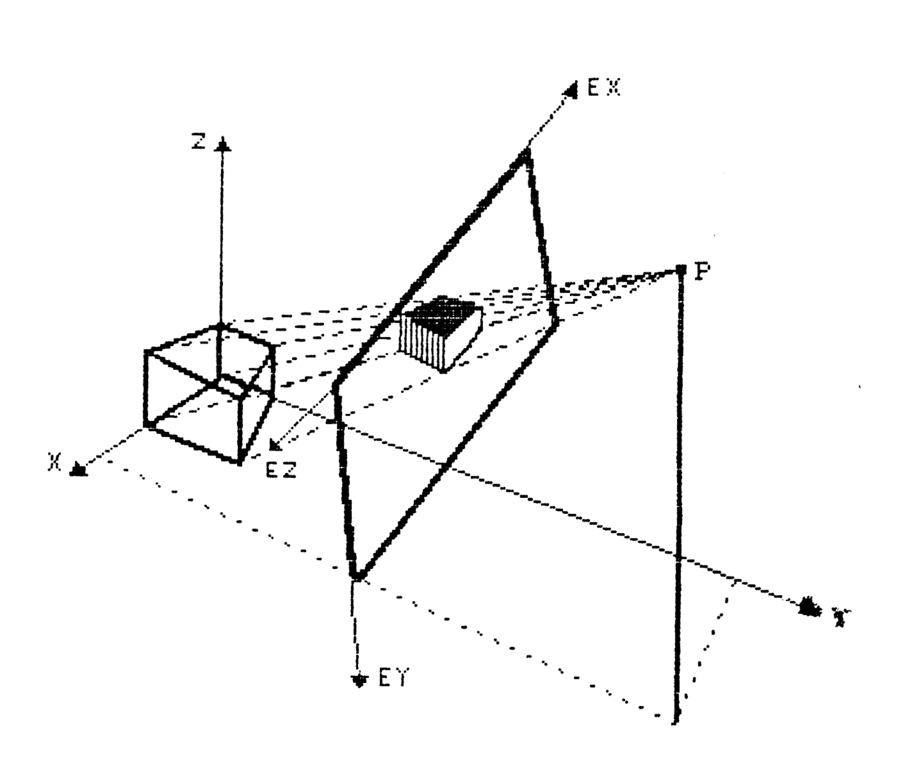


Figure 4.5 — Effets de la projection d'un objet tridimensionnel sur une surface bidimentionnelle. Il est nécessaire d'introduire des déformations pour suggérer l'effet de perspective.

Points de fuite

Par conséquent, comme le but que nous poursuivons est de tromper le cerveau en lui faisant croire que ce qu'il « voit » est la réalité, nous serons obligés de n'utiliser que le type perspective. La figure 4.6 met en évidence ces deux formes de perspectives et justifie le choix effectué. Selon les arrêtes ou les faces de la figure qui se trouveront en parallèle avec le plan de projection ou non, on pourra dire que celle-ci est à un, deux ou trois *points de fuite*, le dernier cas étant celui le plus fréquemment rencontré (objet vu selon un angle de vision quelconque).

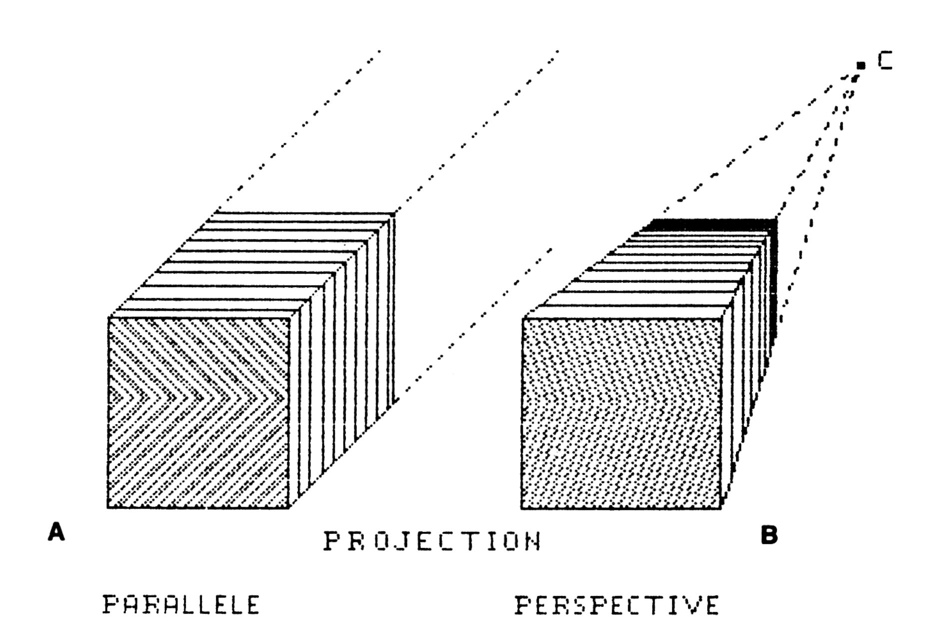


Figure 4.6 — Deux types de projection: en « A » la projection parallèle. En « B » la projection perspective, avec point de fuite. Cette dernière se rapproche davantage de la vue réelle.

Projections parallèle ou perspective

Posons donc dès à présent les bases qui serviront aux projections. Une coordonnée écran correspondra à toute coordonnée d'un point de l'espace. La figure 4.7 montre que l'objet réel sera toujours rapporté à un système d'axes (X, Y, Z) et l'écran sera considéré comme un plan perpendiculaire à une droite (**O O'**) joignant l'origine de ce système d'axes à l'œil de l'observateur. Nous aurons aussi besoin de connaître la distance (**D**) séparant cet œil de l'écran, ainsi que celle (**R**) reliant l'observateur à l'origine du système **X, Y, Z.**

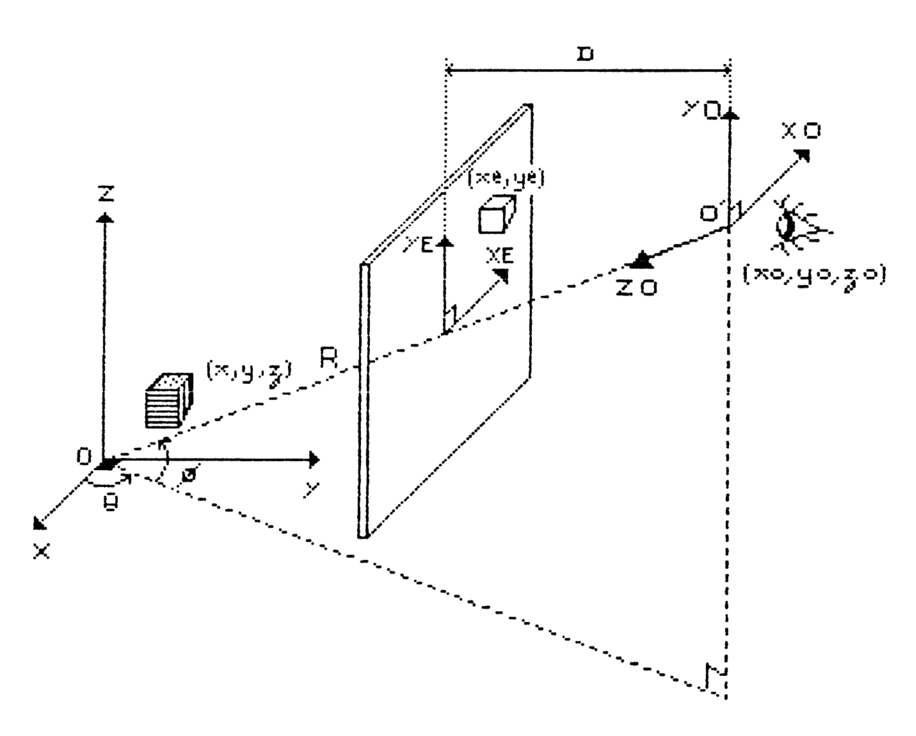


Figure 4.7 — Intersection du plan de l'écran avec les systèmes d'axes tridimentionnels, se rapportant d'une part à l'objet et d'autre part à l'œil.

Coordonnées sphériques

Les différents calculs de projection feront donc intervenir trois systèmes d'axes: celui de l'objet observé (**X, Y, Z**), celui de la position de l'œil dans l'espace (**XO, YO, ZO**) et celui de la correspondance écran (**XE, YE**). Les coordonnées seront obligatoirement sphériques, ce qui permettra la localisation précise de l'observateur dans l'espace, simplement en précisant les valeurs des deux angles θ et φ et de **R**.

θ, φ et ρ

En faisant varier **R**, nous simulerons un éloignement ou un rapprochement de l'objet, qui sera alors perçu comme étant plus petit ou plus grand. Les variations de **D** permettront une simulation d'*effet de zoom* ou de *grand angulaire*, qui produira ou non les déformations liées à ce genre d'objectifs.

Pour tracer la vue en perspective, chaque point de l'objet sera projeté sur le plan de l'écran. Afin de parvenir aisément à ce résultat, qui nécessitera le passage du système d'axes **XO, YO, ZO** à celui **XE, YE**, nous utiliserons les deux relations géométriques suivantes:

$$xe = \frac{D \cdot xo}{zo} \qquad ye = \frac{D \cdot yo}{zo}$$

Projection

qui effectueront les transformations des coordonnées **xo, yo, zo** relatives au système d'axes de l'œil, d'un point quelconque, en ses coordonnées de projection sur l'écran, **xe** et **ye**.

Maintenant que nous avons à notre disposition les connaissances et les outils nécessaires à la représentation des formes en trois dimensions sur un écran, nous allons la réaliser progressivement, par étapes.

TRANSLATION DU SYSTEME D'AXES DE L'OBJET VERS CELUI DE L'ŒIL

Faisons appel à nos connaissances. Nous avions vu que les coordonnées de l'œil par rapport à l'origine des axes **X, Y, Z** pouvaient se déduire par:

Translation

$$xo = R \cdot \cos\theta \cdot \cos\varphi$$
$$yo = R \cdot \sin\theta \cdot \cos\varphi$$
$$zo = R \cdot \sin\varphi$$

La matrice de translation sera donc la suivante:

$$T1 = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -xo & -yo & -zo & 1 \end{vmatrix}$$

ROTATION DU SYSTEME TRANSLATE AUTOUR DE Z'

Rotation

Cette rotation sera effectuée pour un angle **θ−90** dans le sens des aiguilles d'une montre, de façon à ce que le prolongement de **O' Y'** vienne couper l'axe **Z**.

Une remarque s'impose. Jusqu'ici nous avons évalué l'angle φ en prenant sa valeur selon un « écartement » par rapport au plan **X O Y** (en remontant). Il est parfaitement acceptable, bien que moins naturel, de considérer cet écartement par rapport à l'axe **Z** (en descendant). Dans ce cas, les formules servant à calculer les coordonnées de l'œil vont se transformer en:

$$xo = R \cdot \sin \varphi \cdot \cos \theta$$
$$yo = R \cdot \sin \varphi \cdot \sin \theta$$
$$zo = R \cdot \cos \varphi$$

La matrice de rotation appropriée sera :

$$R = \begin{vmatrix} \sin \theta & -\cos \theta & 0 & 0 \\ \cos \theta & \sin \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Et comme nous avons vu qu'il était préférable d'imposer une rotation aux axes plutôt qu'aux points, il vaut mieux en considérer l'inverse :

$$R^{-1} = \begin{vmatrix} \sin \theta & \cos \theta & 0 & 0 \\ -\cos \theta & \sin \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

NOUVELLE ROTATION DE CE SYSTEME AUTOUR DE X'

Rotation inverse

Cette fois pour un angle φ **+90**, dans le sens trigonométrique (inverse de celui des aiguilles d'une montre). Ainsi l'axe **Z'** sera dirigé vers **O** de **X, Y, Z.**

Nous aurons de même :

$$S = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & -\sin \varphi & -\cos \varphi & 0 \\ 0 & \cos \varphi & -\sin \varphi & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

CONVERSION DANS LE SYSTEME HABITUEL

Conversion

Avec un tel système, l'axe des **X'** doit se diriger vers la droite, conventionnellement.

Cette transformation sera automatiquement effectuée en inversant les éléments de la colonne des **X**. Soit :

$$I = \begin{vmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Si le raisonnement a été suivi (il est vrai qu'il était relativement complexe), le système d'axes **X**, **Y**, **Z** transformé en **X'**, **Y'**, **Z'** sera « plaqué » exactement sur celui de l'œil (**XO**, **YO**, **ZO**) et se confondra avec lui. Le but recherché a été atteint par des moyens détournés, à savoir positionner le point de vue correctement, tout en laissant l'objet fixe.

Matrice de transformation générale

Heureusement, l'enchaînement des quatre étapes précédentes peut se simplifier considérablement, pour parvenir à une seule matrice qui est le résultat de leur produit :

$$M = \begin{vmatrix} -\sin\theta & -\cos\theta\cdot\sin\varphi & -\cos\theta\cdot\cos\varphi & 0 \\ \cos\theta & -\sin\theta\cdot\sin\varphi & -\sin\theta\cdot\cos\varphi & 0 \\ 0 & \cos\varphi & -\sin\varphi & 0 \\ 0 & 0 & R & 1 \end{vmatrix}$$

et ainsi $\quad |\, xo \;\; yo \;\; zo \;\; 1 \,| = |\, x \;\; y \;\; z \;\; 1 \,| \times M$

soit:

$$xo = -x\cdot\sin\theta + y\cdot\cos\theta$$
$$yo = -x\cdot\cos\theta\cdot\sin\varphi - y\cdot\sin\theta\cdot\sin\varphi + z\cdot\cos\varphi$$
$$zo = -x\cdot\cos\theta\cdot\cos\varphi - y\cdot\sin\theta\cdot\cos\varphi - z\cdot\sin\varphi + R$$

Les coordonnées **xo, yo, zo** serviront au calcul des coordonnées écran **xe, ye** comme vu plus haut, c'est-à-dire en projection *parallèle*:

$$xe = xo \qquad ye = yo$$

Projection perspective

et en projection *perspective*:

$$xe = \frac{D\cdot xo}{zo} \qquad ye = \frac{D\cdot yo}{zo}$$

Effet de zoom

Un objet pourra alors être affiché avec toutes sortes de modifications. Soit en changeant les valeurs de θ et φ de façon à tourner autour de lui, soit en racourcissant ou en allongeant **R** pour se rapprocher de l'objet ou pour s'en éloigner (avec les effets de déformations générés par la perspective). Et finalement en compensant ces modifications de taille par l'action sur **D**, qui nous servira ici d'objectif photographique interchangeable, ou plus exactement de *zoom*. C'est-à-dire en rapetissant ou en agrandissant l'image finale à volonté, afin qu'elle occupe la surface désirée de l'écran. Et ceci sans modifier l'apparence du tracé dans son ensemble.

Par contre, le résultat que l'on obtiendrait dès maintenant, ne serait pas encore totalement satisfaisant. L'objet serait bien fidèlement reproduit point par point, mais justement *tous* ses points seraient retracés. Or dans la réalité (celle que nous cherchons à obtenir à tout prix) un solide comporte des surfaces ou des lignes cachées par d'autres et ceci en fonction du *point de vue* de l'observateur.

4. L'ELIMINATION DES SURFACES CACHEES

Si le problème des surfaces cachées à éliminer du tracé peut paraître simple à un dessinateur, effectuer la même tâche au niveau d'un dessin informatique et ceci *automatiquement*, est non seulement complexe mais très coûteux en temps de calculs et en espace mémoire. Des algorithmes très variés existent de nos jours, chacun possédant ses qualités et défauts qui lui sont propres. Il nous faut donc simplifier le problème et ne conserver qu'un procédé simple, plus ou moins universel, mais qui soit suffisamment performant.

Il est possible de regrouper les différents cas que nous serons amenés à rencontrer, en deux catégories principales:

— les objets solides à afficher sous forme de surfaces délimitées par des lignes.

— les fonctions représentées par des séries de lignes formant un « maillage », croisé ou non.

LA METHODE DES SURFACES CACHEES

Elle peut être appliquée lors des visualisations d'objets, qu'il s'agisse de cubes, pyramides, maisons, avions, formes techniques ou autres.

Surfaces cachées

Des méthodes de ce type ont été mises au point par des chercheurs, avec plus ou moins de complexité au niveau des calculs et plus ou moins de perfectionnements quant aux résultats obtenus. Une méthode simple consistait à dessiner les objets échelonnés dans divers plans, en commençant par le plus éloigné. Le principe retenu était équivalent à celui concernant les surfaces cachées dans le domaine bi-dimensionnel: le dessin de la surface supérieure recouvre et cache en partie celui qui lui est immédiatement inférieur.

Mais évidemment, au sein de l'espace tri-dimensionnel, le procédé était plus complexe. Il fallait d'abord connaître les coordonnées du dessin situé plus en avant dans le plan horizontal, effacer les parties communes en dessinant des lignes de la couleur du fond aux emplacements désirés, en superposition sur le tracé inférieur et enfin tracer la dernière figure. Ainsi même si les objets montrés ne comportaient que des lignes délimitant des surfaces, l'effet d'effacement des surfaces cachées était parfaitement simulé.

Une telle solution, pour satisfaisante qu'elle paraisse être lors du tracé de deux ou trois objets simples, ne peut pas nous contenter pleinement. En effet, nous désirons obtenir des tracés automatiques nous déchargeant entièrement du souci de générer telle ou telle action spécifique en fonction de tel cas particulier. L'objet de ce livre étant la vulgarisation des méthodes graphiques évoluées et leur application au relief, nous ne nous attarderons pas non plus sur les résultats obtenus de nos jours par les meilleurs « artistes peintres en informatique ». Les productions actuelles remplissent les pages en polychromie des revues spécialisées. Certaines ont atteint un niveau de qualité proche de la perfection. Mais elles nécessitent des matériels *haut de gamme* spécialisés et valant fort cher, ainsi qu'un recours massif à l'*assembleur.*

Evidemment, ces machines sont capables de créer de superbes images complexes à la cadence de *30 vues par seconde.* Mais les instructions graphiques qu'elles comportent sont *câblées* au sein du processeur et ne nécessitent même pas le recours à l'assembleur (du moins en ce qui concerne les *primitives* du langage de dessin). Par contre le dessinateur ne connait généralement pas les différents algorithmes qu'il utilise.

Turbo Pascal

Pour notre part, nous désirons surtout comprendre les mécanismes utilisés par tous, afin de pouvoir les adapter facilement et surtout posséder les bases nécessaires à des améliorations futures. De plus ces techniques doivent être mises en oeuvre sur des micro-ordinateurs variés, couvrant la gamme des « familiaux » aux « professionnels ». Les méthodes utilisées devront donc obéir à une certaine standardisation, incluant le recours soit au **BASIC**, soit au **Turbo Pascal** pour le haut de gamme.

Vecteurs de visibilité

Compte tenu de ces considérations, le problème des surfaces cachées peut être résolu d'une façon assez satisfaisante par la méthode du test de visibilité, faisant appel à la notion de *vecteurs.* Toutefois ces surfaces doivent concerner un objet **convexe**. Afin de déterminer quelles surfaces sont cachées à la vue de l'observateur, il peut être judicieux de les délimiter par des *vecteurs*, c'est-à-dire par des segments orientés comportant une *origine* et une *extrémité.*

C'est par le produit de deux vecteurs que nous serons renseignés sur le fait qu'une face est située en avant ou en arrière par rapport au point d'observation. Les mathématiques vont en effet, une fois de plus, se porter à notre secours. Elles enseignent que le produit vectoriel de deux vecteurs est égal à un vecteur orienté *perpendiculairement* au plan qu'ils forment. Naturellement nous sommes préocupés par le fait que ce troisième vecteur puisse pointer vers l'ex-

Numérotation trigonométrique

térieur (vers l'observateur) ou l'intérieur de l'objet. Comme seule la première solution nous est utile, nous serons obligés d'orienter nos vecteurs selon un sens bien déterminé, lequel sera défini en numérotant les sommets des faces dans le sens *trigonométrique*; le point de départ étant pris arbitrairement.

La figure 4.8 (A) permet de mieux saisir ces assertions. Nous avons numéroté la première face du cube dans le sens trigonométrique, en débutant par le coin en bas à droite. Tout autre point de départ aurait tout aussi bien pu convenir.

Produit vectoriel

Puis nous traçons deux vecteurs, inclus dans le plan de cette face, ayant pour origine commune le coin « 1 » et orientés respectivement vers les coins « 2 » et « 3 ». Si maintenant nous effectuons leur produit vectoriel, le résultat obtenu sera formé par un nouveau vecteur $\vec{P}$, perpendiculaire à la face considérée et orienté dans notre direction. Enfin le principe du produit scalaire de deux vecteurs nous permettra de savoir si cette face est visible ou non, du point de vue considéré. Ainsi :

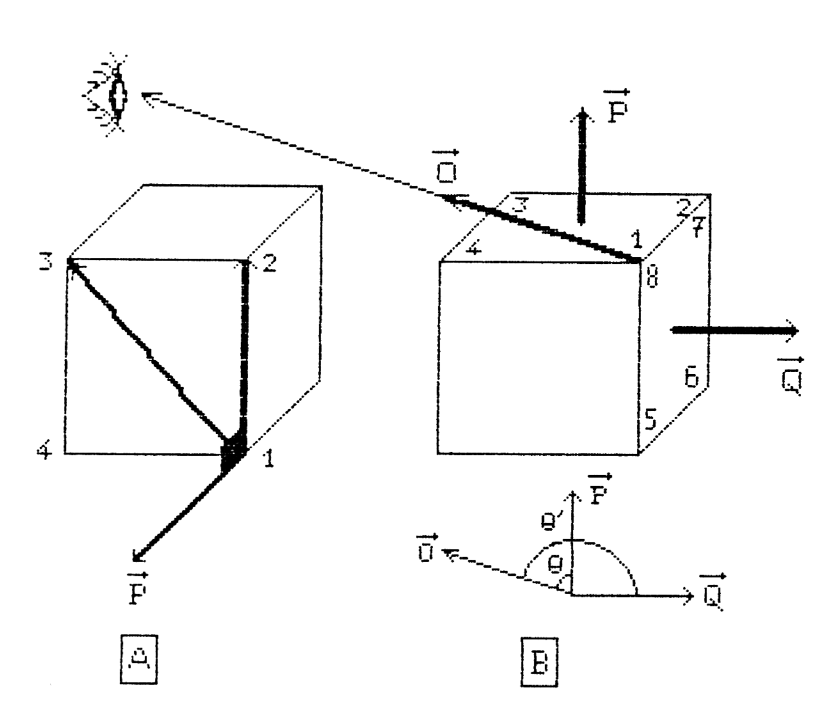


Figure 4.8 — Le produit vectoriel (en « A »), permet de déterminer si une face de l'objet est cachée ou non de l'observateur (en « B »).

$$\vec{2}\cdot\vec{3} = |\vec{2}|\cdot|\vec{3}|\cdot\cos\theta$$

θ est l'angle formé par les deux vecteurs $\vec{2}$ et $\vec{3}$. Or, mathématiquement, le signe d'un tel produit scalaire ne dépend que de **COS** θ. Par voie de conséquence, ce produit ne sera positif qu'à condition que **COS** θ le soit aussi, donc que θ soit compris entre 0 et 90 degrés. Inversement il sera négatif lorsque θ sera compris entre 90 et 180 degrés.

Test de visibilité d'une face

Maintenant, pour tester avec certitude la visibilité d'une face, il convient d'introduire un nouveau vecteur ayant pour origine l'un des sommets de la face considérée et pour direction celle de l'œil qui l'observe.

La figure 4.8 (B) démontre son utilisation. $\vec{P}$ est le produit vectoriel de $\overrightarrow{1\,2}$ et $\overrightarrow{1\,3}$ par exemple; $\vec{Q}$ celui de $\overrightarrow{5\,6}$ et $\overrightarrow{5\,7}$. Ils sont perpendiculaires à leur face et orientés vers l'extérieur du cube. Ensuite nous construisons $\vec{O}$. Selon le principe du produit scalaire de $\vec{O}\cdot\vec{P}$ d'une part et de $\vec{O}\cdot\vec{Q}$ d'autre part, nous pouvons voir, d'après les vecteurs reportés sous le cube, que θ est inférieur à 90 degrés mais que θ', lui, est supérieur. Par conséquent la face 1 2 3 4 sera visible à la position de l'observateur, tandis que la face 5 6 7 8 lui sera cachée. Ce qui parait évident sur ce dessin.

LA METHODE DES LIGNES CACHEES

Lorsqu'une image est représentée par des lignes, c'est-à-dire lors d'un « maillage » représentant la visualisation d'une fonction en trois dimensions par exemple, l'application de la méthode précédente n'est plus possible. Il est nécessaire, à ce niveau, de se préocuper des *lignes de crête* formées.

Lignes de crête

Ceci est rendu possible par le fait que ce type de tracé est obtenu par un découpage d'une fonction en tranches, séparées les unes des autres par la largeur d'un « pas » constant. Chaque tranche sera matérialisée par des points dont les coordonnées varieront selon la fonction traitée. Il faut commencer par tracer la tranche la plus proche de l'observateur, puis celles qui s'en éloignent, graduellement. Un cas particulier se présentera alors rapidement, lorsqu'une ligne de crête passera au-dessous de celle dessinée auparavant.

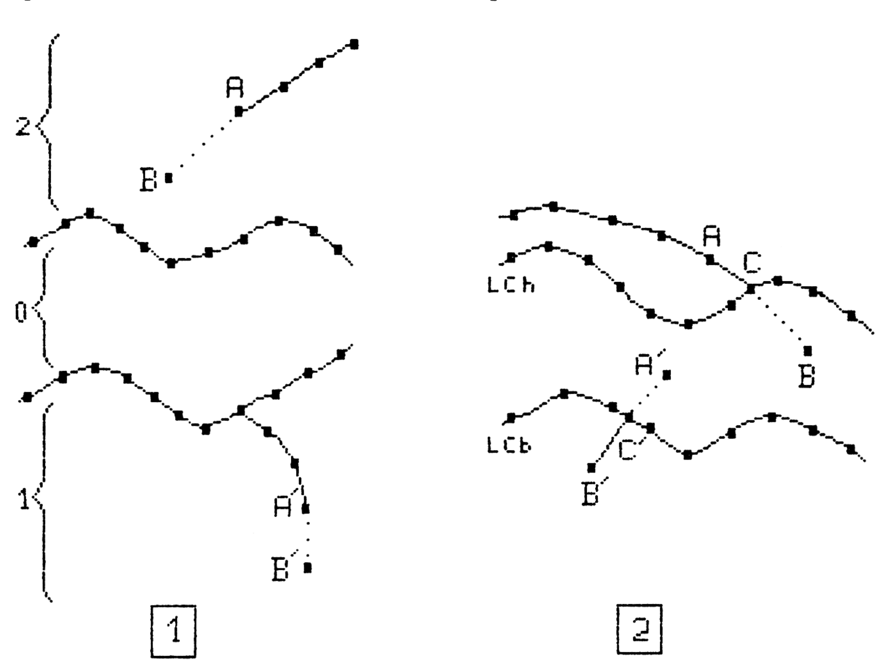


Figure 4.9 — En « 1 », division de l'écran en trois zones (0, 1 et 2). En « 2 », détermination des parties cachées d'une ligne, selon le franchissement des lignes de crête.

Ce procédé suppose la tenue à jour continuelle de *lignes de crêtes* représentant les tracés les plus « hauts » et les plus « bas » effectués sur l'écran, jusqu'à maintenant. L'écran peut alors être divisé en trois zones, **1**, **0** et **2** selon leur situation par rapport à ces deux lignes (voir figure 4.9). Considérons que **A** concerne le dernier point tracé et **B** celui qui doit être tracé ensuite. Quatre cas peuvent alors se présenter:

Trois zones de visibilité

1) **A** et **B** appartiennent au même code, autre que **O** (cas 1 de la figure 4.9). Par exemple A=2 et B=2, A'= 1 et B'=1. Ceci implique que le segment **A B** (ou **A' B'**) est visible. Il sera donc tracé.

2) **A** et **B** sont situés ensemble dans la zone **O**. Si A=O et B=O, cela signifie que le segment **A B** est complètement invisible et ne devra pas être tracé.

3) **A** et **B** sont compris chacun dans une zone différente, mais contiguës. C'est le cas en 2 de la figure 4.9. A=2 et B=O, A'=O et B'=1 et leurs inverses. Le procédé se complique, car seule une partie du segment de droite **A B** (ou **A' B'**) est visible et doit être tracé. Il faut donc impérativement calculer les coordonnées du point **C** (ou **C'**), situé à l'intersection d'une ligne de crête et du segment considéré. Ainsi seul **A C** sera tracé, ou **C' B'**. Pour déterminer les coordonnées de **C** il faudra calculer:

$$CX = \frac{AY \cdot BX - BY \cdot AX - LC(AX) \cdot BX + LC(BX) \cdot AX}{LC(BX) - LC(AX) - BY + AY}$$

$$CY = \frac{AY \cdot LC(BX) - BY \cdot LC(AX)}{LC(BX) - LC(AX) - BY + AY}$$

X et **Y** étant les coordonnées de **A**, **B** ou **C** et **LC** une table des valeurs **Y** de la ligne de crête considérée, pour chaque **X** de l'écran.

4) **A** et **B** appartiennent à deux zones de code différent, mais non contiguës. Par exemple A=2 et B=1 et inversement. Dans ce dernier cas le segment **A B** coupe les deux lignes de crête et seules ses parties supérieures et inférieures sont visibles et doivent être tracées. Il sera donc nécessaire de calculer **C** et **C'**, points d'intersections de **A B** avec chaque ligne de crête. Le tracé portera sur **A C** et **C' B** selon la même méthode que celle expliquée ci-dessus, mis à part le fait que les coordonnées de deux points sont à déterminer, au lieu d'un seul.

Interpolation linéaire

Pour calculer régulièrement les nouvelles lignes de crête dont les valeurs sont rangées dans des tables, on utilisera la méthode de l'interpolation linéaire de la forme **y=ax+b**, qui permettra de trouver les valeurs **Y** pour chaque **X** de l'écran compris entre **A** et **B** (cas 1 de la figure 4.9 dans lequel **A B** devient la nouvelle ligne de crête supérieure):

$$y = \frac{(BY - AY) \cdot x + BX \cdot AY - AX \cdot BY}{BX - AX}$$

Voici donc tous nos problèmes résolus. Les techniques étudiées jusqu'ici seront appliquées méthodiquement dans nos programmes d'application constituant la deuxième partie de ce livre. Nous sommes maintenant capables de visualiser un objet ou une surface complexe en trois dimensions, selon n'importe quelle position de l'observateur dans l'espace, selon la distance à laquelle il est sensé se trouver et avec un effet de perspective, calqué sur celui observé réellement. Nous sommes prêts à bâtir des programmes capables de faire comprendre simplement et automatiquement à des ordinateurs, les concepts que nous venons d'élucider.

Pourtant à ce stade nous serons encore déçus: il y aura encore une différence entre ce qui sera affiché et ce qui serait observé dans la réalité; ce qui est observé est irrémédiablement « plat », même si la notion de perspective suggère un certain *relief.* Il reste donc un dernier effort à accomplir au niveau de la théorie, pour que le cerveau de l'utilisateur soit enfin « trompé » totalement et perçoive *réellement* le **relief** du tracé affiché.

5. LE RELIEF ! (A CHAQUE ŒIL SON IMAGE)

Le véritable relief

Le chapitre 3 avait montré que la perception du relief était un phénomène *subjectif.* Il avait été démontré l'importance de la vision binoculaire pour sa mise en oeuvre. Or des procédés existent, au niveau de la photographie, afin de recréer cette sensation de *relief.* Partant de ces principes, il est possible de les remodeler, en vue d'obtenir les mêmes effets dans le domaine informatique. Passons donc en revue ceux applicables à la prise de vue photographique, avant d'étudier leur adaptation sur nos « ordinateurs ».

LA PHOTOGRAPHIE EN RELIEF

Deux images différentes

Le secret de la vision en relief consiste donc à faire observer, par chaque œil, une image différente de celle observée par l'autre. Mais cependant pas de n'importe quelle manière. Il est de la plus haute importance de recréer avec un maximum de fidélité, les conditions de la perception binoculaire. Ce qui veut dire que chaque œil doit voir un même objet sous un angle légèrement différent. Chacun a pu constater que lorsqu'on ferme un œil, puis l'autre, des modifica-

Synthèse des images

tions d'emplacement des différentes formes constituant notre environnement, sont sensibles. Ceci est d'autant plus réaliste que les objets sont plus rapprochés de l'observateur. Les deux images perçues ne pourraient pas être superposées exactement, mais le cerveau est capable d'en effectuer une synthèse et de nous faire « sentir » le relief. Lorsque nos deux yeux sont ouverts, nous percevons parfaitement les distances. Si nous fermons un œil cette sensation disparait. Elle n'est plus que déduite, avec plus ou moins d'exactitude.

Stéréoscope

Daguerréotype

C'est au XIXème siècle que des savants britanniques eurent l'idée géniale de construire des *visionneuses* permettant de donner du *relief* à des images photographiques. Le premier « stéréoscope » était né (du grec *stéréos* : solide et *skopein*: voir). Il était aussitôt suivi du « Calotype » et du « Daguerréotype », qui fit fureur à l'époque. On en trouve d'ailleurs encore aujourd'hui des exemplaires dans les greniers et les ventes aux enchères. Les photographies étaient réalisées évidemment en *monochromie*, sur plaque de verre. Curieusement, ce procédé sombra pratiquement dans l'oubli par la suite et ce n'est que tout récemment que des amateurs en reprirent le principe, en lui ajoutant la *couleur.*

Lunettes

Il y a peu de temps même, la télévision tenta la présentation d'un film en relief, avec emploi de lunettes spéciales. Ce fut un relatif échec, car les conditions de retransmission des ondes télévisées diminuaient énormément l'effet recherché. Pourtant le principe reste très valable, même s'il est exploité sous plusieurs formes. Les personnes ayant eu la chance d'acheter le numéro du 5 juillet 1985 de « PARIS MATCH » auront largement pu s'en convaincre. De même « Sciences & Vie », en janvier 1986, présentait de superbes vues en relief de certaines régions de france, photographiées par satellite. Il est très important de conserver précieusement les lunettes spéciales fournies avec ces périodiques; elles seront particulièrement utiles par la suite.

Photographies en relief

En guise de « cadeau », voici comment obtenir de superbes **photographies en relief** avec du matériel ordinaire. Le principe est simple: il consiste en deux prises de vues, décalées de quelques centimètres par l'intermédiaire d'un objectif spécial, ou par deux appareils ordinaires couplés mécaniquement ou électriquement et montés sur un support à glissière. Si les vues ne comportent pas d'objets en mouvement (paysages), un seul appareil au format « 24 x 36 » par exemple, peut convenir. Il suffit d'effectuer deux prises de vues en les décalant d'environ 7 à 10 cm. Augmenter cette distance accentue l'effet de relief, en lui conférant même un « effet de maquette » pour un écartement de plusieurs dizaines de mètres. Deux prises de vues successives, à partir d'un avion, permettent de « voir » le relief d'un massif montagneux. Les focales courtes réalisent les meilleurs effets.

Bien entendu, il est important, lors du déplacement des appareils photographiques, de « fixer » un même point au centre du viseur. En général le déplacement doit être d'environ 1/30ème de la distance de l'objet le plus rapproché. Le film employé sera ordinaire, sur diapositives. La suite de ce livre montrera comment visualiser ces couples de photos en relief. Il ne s'agit pas d'un *gadget.* les résultats obtenus sont réellement spectaculaires, surtout si l'on a pris soin de prévoir des *plans de vue* échelonnés, avec une bonne *profondeur de champ.*

LE RELIEF SUR MICRO-ORDINATEUR

Vrai relief sur micro-ordinateur

Le principe né de la photographie peut être utilisé avec profit pour créer des images en relief, à l'aide de micro-ordinateurs. Les principes de visualisation peuvent diverger, comme on le verra par la suite, mais la méthode de construction des images sera toujours identique. C'est donc par elle que nous allons débuter.

D'après les études précédentes, nous pouvons tracer des surfaces en *trois dimensions* ou des objets *tri-dimensionnels,* en fonction du **point de vue** de l'observateur, dans l'espace. Il importe seulement de définir la distance de l'obser-

Deux tracés tridimensionnels

vateur au sujet, la distance écran et, surtout, les angles θ et φ. Le logiciel devrait pouvoir effectuer le tracé qui en résulte, automatiquement. Ceci en vue d'obtenir **un** dessin en trois dimensions. Or nous désirons en obtenir **deux**, selon le principe des prises de vues photographiques *stéréoscopiques.*

Modification d'angle

Ce deuxième tracé devra donc être en tous points identique au premier, mis à part la simulation du déplacement latéral de « l'appareil de prise de vues ». Le logiciel devra donc être capable de dessiner deux fois le même motif, sous certaines conditions, en simulant le déplacement du point de vue selon le principe du 1/30ème de la distance. En général, une modification de l'angle θ d'environ **5** degrés, **dans le bon sens**, est suffisante. Le programme effectuera les différents calculs nécessaires, pour que les deux tracés soient rigoureusement identiques, à part la modification de θ de 5 degrés. Et cela *automatiquement,* bien entendu.

Procédés de restitution

Par contre, au niveau de la restitution du relief, les procédés peuvent diverger selon le matériel possédé. Plusieurs solutions sont envisageables. La meilleure consiste à obtenir l'effet de relief directement sur l'écran pour être capable de le percevoir en *temps réel.* Pour cela, la possession d'un micro-ordinateur comportant un mode haute résolution graphique couleur, avec deux pages H.R. sélectionnables et la possibilité d'attribuer une couleur indépendante à chaque point de l'écran, est la solution idéale. C'est le cas par exemple d'un TRS-80 modèle I ou III équipé de la haute résolution de « Micro-Influx », ou de la toute dernière norme « MSX-2 ».

IBM-PC et familiaux

Avec un *Apple II* ceci est aussi possible, puisque deux pages H.R. sont sélectionnables, à l'exception près que seules deux couleurs par groupe de 8 points sont permises. Ce n'est pas très gênant, comme nous le verrons plus tard. Un IBM-PC ou compatible est capable de générer de superbes vues, mais il ne comporte qu'une page H.R. en standard. Il est pourtant possible de trouver une excellente solution à ce problème, en utilisant « Turbo Pascal » par exemple. Enfin tous les familiaux ordinaires pourront recréer l'effet de relief, même s'ils ne comportent qu'une page H.R. et la limitation de 2 couleurs seulement par groupes de 8 points (MSX-1, Amstrad, Commodore, pour ne citer que les plus répandus).

La mise en oeuvre des différentes solutions s'effectuera à la fin de la deuxième partie de ce livre. Toutes sont valables, il est simplement important de souligner que certaines sont plus pratiques d'emploi que d'autres. Citons, à titre d'exemple, le cas d'une animation sur l'écran, mais vue, **pour la première fois** comme si un objet solide se déplaçait **réellement** dans l'espace, devant le spectateur *émerveillé* !

CHAPITRE 5

LE LOGICIEL

Le moment est maintenant arrivé de mettre en pratique les enseignements des chapitres précédents. Deux sortes de logiciels vont pouvoir être développés, selon les buts à atteindre: ceux relatifs à la représentation de **fonctions** ou bien les programmes concernant la visualisation d'**objets**. Leur utilisation sera simple et universelle, adaptable à toutes sortes de *périphériques* ou *machines.*

1. LA REPRESENTATION DE FONCTIONS

BASIC MICROSOFT

C'est le programme 5.1 (page 51) qui est chargé de représenter les fonctions en trois dimensions, selon les désidérata de l'utilisateur. Il a été écrit en traditionnel **BASIC Microsoft**. Les seules instructions pouvant varier, en fonction du micro-ordinateur employé, sont regroupées en tête du programme, au sein du « bloc des commandes spécifiques » (lignes 110 à 160).

Son utilisation est très simple. Lors du lancement du logiciel, un message indique que la fonction à traiter doit être écrite à la ligne 240, puis qu'un « run 300 » débutera normalement l'exécution à partir de la ligne 300. A partir de cet instant, un certain nombre de questions vont être posées. Tout d'abord, il conviendra de répondre par « E » lorsque le périphérique de visualisation sera de type *écran*, ou par « T » dans le cas d'une *table traçante.* Puis l'utilisateur va devoir définir une fenêtre de visualisation: le tracé sera toujours compris au sein de ces limites, qu'il occupera au maximum. Ainsi la représentation graphique peut être effectuée à n'importe quel emplacement d'un écran et à une échelle quelconque. Il s'agit d'un cadre limitant la surface exploitable par les coordonnées de ses bords.

Cadre de présentation

Il est possible de provoquer le tracé d'un cadre optimisé, autour du dessin, de la couleur désirée. Un message d'erreur a été prévu au cas où ce cadre déborderait les limites « physiques » de l'écran. Un bon principe est de définir la fenêtre 6 *pixels* à l'intérieur de la surface souhaitée. La couleur du tracé est aussi demandée; nous verrons plus loin que ce point est très important.

Limites de l'étude

Viennent ensuite les *bornes* de la fonction. C'est-à-dire les limites de l'étude, à prendre en compte pour sa représentation. De ce fait il est possible de visualiser la fonction sur une surface étendue, ou seulement en un endroit précis. Par souci de symétrie, ces bornes sont généralement choisies de part et d'autre des axes de coordonnées: par exemple de -5 à $+5$ sur l'axe des « x » et des « y ».

RHO conditionne l'aspect final

Il est encore nécessaire de connaître le nombre de lignes à tracer, ainsi que le nombre de points par ligne, sachant que plus ces nombres sont élevés, plus le tracé sera fin et précis, mais plus le temps d'exécution sera important. D'autre part, il est primordial de les choisir parmi les nombres premiers. Ceci afin d'éviter que les calculs passent par un *pôle* de la fonction et ne provoquent une grave erreur. La distance « RHO », à laquelle est sensé se trouver l'observateur, est ensuite demandée. Elle conditionnera non seulement la taille du dessin, mais aussi son aspect final. Par contre la distance de l'écran influencera l'aspect du tracé, c'est-à-dire avec un effet de *grand angulaire*, de *focale normale* ou de *télé-objectif.* En général, une distance de **1** est un bon choix et procure une vision ordinaire. Enfin les deux dernières questions sont les plus importantes. Elles définissent la position de l'*observateur* dans l'espace par rapport à la fonction. « THETA » influence le déplacement horizontal, autrement dit l'angle d'*azimut.* « PHI » concerne quant à lui le déplacement vertical, c'est-à-dire l'angle de *site.*

Délais variables

Les questions sont à présent terminées et le programme peut maintenant commencer son traçage, avec des délais s'étendant de 5 minutes à 1 heure, selon la précision demandée. Par contre tout est automatisé, de la représentation *réelle* dans l'espace en fonction du *point de vue*, jusqu'à l'*élimination automatique des lignes cachées.* La figure 5.1 montre les effets des différents choix et les écueils à éviter. Quatre fenêtres ont été définies sur un écran, avec pour chacune des choix différents. La fonction définie à la ligne 240 est celle répertoriée à la ligne 220. La première vue (5.1a) avait comme spécifications :

- bornes **x,x'** et **y,y'** de la fonction : −8 à 8.
- nombre de lignes à tracer et nombre de points par ligne : 19.
- **RHO**: 8
- distance écran : 1.
- **THETA**: 60 degrés.
- **PHI**: 30 degrés.

Cela s'améliore.

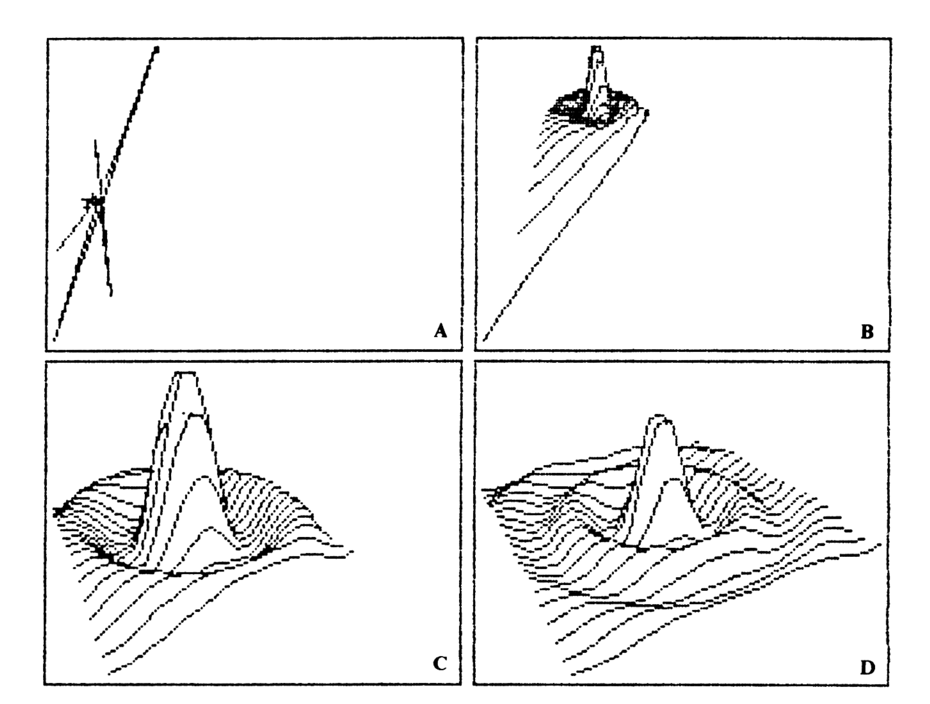


Figure 5.1 — Représentation d'une fonction mathématique, en trois dimensions, avec élimination automatique des lignes cachées. En « A », l'observateur est situé trop près de l'écran (RHO=8). En « B », l'éloignement est plus sensible (RHO=10). En « C », l'effet est encore plus accentué (RHO=20). Enfin la distance idéale est représentée en « D », avec RHO=40.

On ne peut vraiment pas affirmer que le résultat soit à la hauteur des espérances! L'utilisateur est même en droit de se demander si une erreur ne s'est pas produite à un moment donné. En réalité il n'en n'est rien et la figure 5.1b le montre bien. Tout en conservant les mêmes paramètres, il a suffit d'indiquer:

RHO=10

Choix des paramètres

pour qu'un semblant de représentation apparaisse. Mais l'observateur est encore situé trop près du dessin, ce qui provoque cet effet de fuite à l'horizon. Le tracé en 5.1c est optimisé par:

RHO=20.

Enfin, le dessin en 5.1d représente la même fonction, mais avec les bornes **x,x'** et **y,y'** s'étendant de −13 à 13. La vue étant élargie, il a fallu définir **RHO**=40, afin d'en obtenir une représentation agréable. Cet exemple montre bien l'importance du choix des différents paramètres et des essais successifs qu'il est souvent utile d'effectuer, avant d'arriver à un résultat satisfaisant. En pratique, on se contentera d'un tracé grossier mais rapide (nombre de lignes et nombre de points par ligne = 7), destiné à dégrossir le problème. Un dernier tracé, avec une précision de 43 par exemple, génèrera alors un superbe tracé.

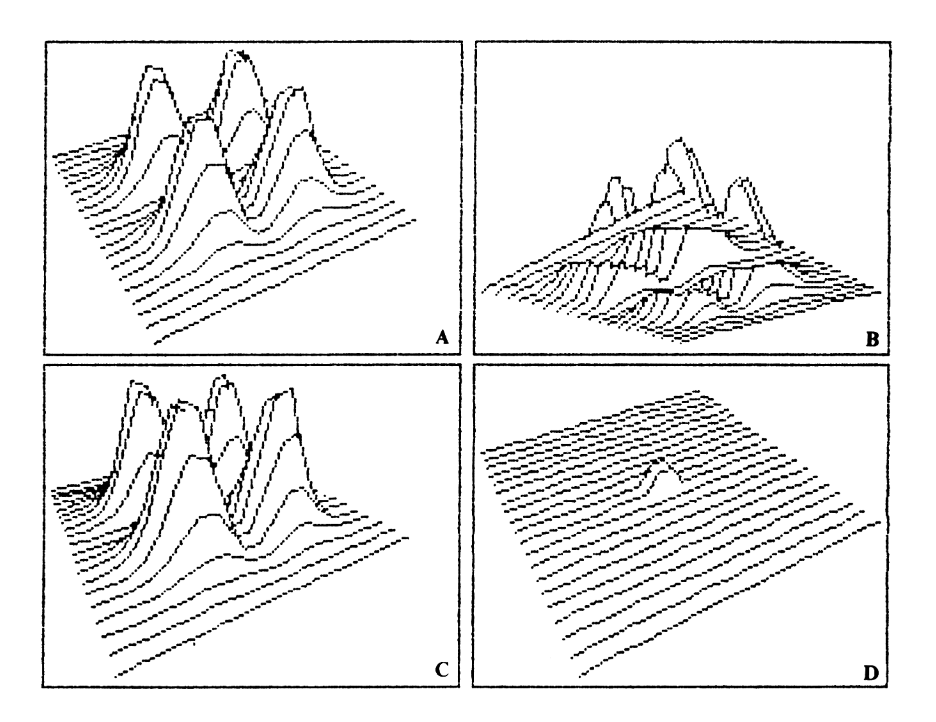


Figure 5.2 — Une autre fonction mathématique représentée en trois dimensions. La position de l'observateur dans l'espace, ainsi que le choix des différents paramètres de vision, conditionnent le résultat final. En « A » et « B » seuls les angles THETA et PHI diffèrent. En « C » un rapprochement a été introduit, par diminution de RHO. Enfin en « D », les bornes de la fonction ont été élargies.

Le programme suggère des valeurs.

La figure 5.2 montre l'influence de la position de l'observateur dans l'espace, par rapport au dessin. En 5.2a, le programme a été lancé avec les paramètres indiqués par défaut. La seule différence concerne l'écran, qui a été divisé en quatre fenêtres. Les paramètres étaient donc:

RHO=8
THETA=60
PHI=40

Toutes les bornes de la fonction étaient comprises dans l'intervalle −2, 2. La fonction est donc vue du dessus et à droite. La figure 5.2b a conservé les mêmes paramètres, sauf pour **THETA=−45** et **PHI=−20**. Nous voyons donc le dessin par la gauche et du dessous.

Déplacements autour de la fonction

En ce qui concerne la figure 5.2c, **THETA=60**, **PHI=30** et **RHO = 5**. Par rapport à la figure située immédiatement au-dessus, le point de vue a légèrement baissé, mais surtout l'observateur s'est rapproché de la fonction. L'effet de perspective est ainsi davantage marqué.

Enfin, la dernière vue, 5.2d conserve les paramètres de la première (5.2a), mais ici les bornes sont comprises entre −13 et 13 et **RHO** est égal à 40. L'observateur s'est éloigné de la fonction et en embrasse une vue élargie. Notre chaîne de montagnes est donc bien isolée, au milieu du désert!

Il convient maintenant de détailler le programme 5.1, qui est à l'origine de ces tracés. Le bloc des commandes spécifiques procure au logiciel une *excellente portabilité* . Il pourra être exécuté sur la quasi totalité des micro-ordinateurs modernes, simplement en modifiant les lignes 110 à 160.

Haute résolution graphique

La ligne 110 efface l'écran en *basse résolution.* La ligne 120 indique la surface de l'écran *haute résolution* (320 pixels en largeur sur 250 pixels en hauteur). La ligne 130 provoque une visualisation en mode *texte,* tandis que la ligne 140 est relative au mode *haute résolution graphique.* Quant aux lignes 150 et 160, elles rappellent pour mémoire de quelle façon indiquer la couleur du tracé qui va suivre et comment tracer un segment de droite. Chaque utilisateur aura ainsi la possibilité d'adapter simplement le programme à son type de matériel.

HCOLOR et HPLOT

Il est vraiment difficile de prévoir plus simple, les ordres spécifiques étant réduits à leur strict minimum. En ce qui concerne **HCOLOR** et **HPLOT**, il conviendra d'en modifier la syntaxe dans le corps du programme (si vous disposez d'un éditeur de textes, vous pourrez utiliser la commande « find » à cet effet). Cette option a été prise afin de ne pas pénaliser le temps d'exécution par des appels trop répétés à des sous-programmes.

Les lignes 190 à 220 indiquent simplement quatre types de fonctions possibles. L'une d'elles devra être recopiée en ligne 240. Il est loisible d'en ajouter bien d'autres, comme par exemple:

```
Z = 3*((SIN(X)*SIN(X))*SIN(Y)
Z = - 8*EXP( - X*X - Y*Y)*(X + Y)
Z = 7*EXP( - 0.1*(X*X + Y*Y))
```

Pyramide de vision

Les lignes 240 à 270 représentent un sous-programme définissant la *pyramide de vision* par rapport à l'écran et à la fonction. Puis le programme principal commence à la ligne 300 et débute par l'entrée des différentes variables, comme nous l'avions vu précédemment. Le type de visualisation « Ecran » indique une orientation de l'axe des **Y** du haut vers le bas et inversement pour le type « Table traçante ».

Les limites de la fenêtre signifient: bord situé à « n » points à partir de la gauche ou du haut. En ligne 400, lorsqu'un cadre est souhaité, il a été prévu de laisser un léger espace entre son pourtour et le tracé de la fonction. Ceci afin que le résultat final soit plus esthétique. A titre d'exemple, chaque question indique entre parenthèses les réponses souhaitables, avec une liste des nombres premiers possibles, pour les lignes 450 et 460.

Il est plus rapide de modifier une ligne de programme que de créer une fonction d'analyse syntaxique.

```
10 '**********************************************************
20 '******* EQUATIONS: SURFACES EN 3D  (II11PGM/BAS)  *******
30 '**********************************************************
40 '
50 GOSUB 130: GOSUB 110 'Effacement ecran B.R.
60 PRINT "Ecrire la fonction en ligne 240 (240 Z= .....)": PRINT "Puis faire:  RUN 300": PRINT:P
RINT: STOP
70 '
```

```
80 '            =================================
90 '===========  bloc des commandes specifiques  ============
100 '           =================================
110 CLS: RETURN 'Effacement ecran Basse resolution
120 MX=320: MY=250: RETURN 'Surface ecran H.R. (Nbe de points)
130 HDISP 1: RETURN 'Visu Basse Resolution
140 HDISP 2: RETURN 'Visu Haute Resolution
150 'HCOLOR x  => Couleur du trace (0 a 7)
160 'HPLOT x,y TO x',y'  => Trace d'une droite
```

Bloc à adapter selon le type de votre machine.

Quelques fonctions toutes prêtes, pour la ligne 240.

```
170 '=========================================================
180 '
190 'Z= 80+60*(SIN(.053*SQR(X*X+Y*Y))):'*****  CORNET
200 'Z= SIN(SQR(X*X+Y*Y)):'*****  CHAPEAU
210 'ZZ=EXP(-X*X-Y*Y)*SIN(X)*SIN(Y):Z=80*(ZZ*ZZ):'*** 4 BOSSES
220 'ZZ=SQR(X*X+Y*Y): Z= 10*SIN(ZZ)/ZZ:'***** VAGUES
230 '
240 ZZ=EXP(-X*X-Y*Y)*SIN(X)*SIN(Y):Z=80*(ZZ*ZZ)
250   OX= (-X*W1)+(Y*M1): OY= (-X*W3)-(Y*W4)+(Z*M2): OZ= (-X*W5)-(Y*W6)-(Z*W2)+RHO
260   XA= D*OX/OZ: YA= D*OY/OZ
270 RETURN
280 '
```

C'est ici que la fonction est définie.

Attention, on commence !

Le sens du tracé peut différer.

```
290 '******************  Debut du programme  ******************
300 GOSUB 120: GOSUB 130: GOSUB 110 'Effacement ecran B.R.
310 INPUT "Type de visualisation : <E>cran ou <T>able tracante "; TT$
320 INPUT "Limite GAUCHE fenetre (6) ";F1
330 INPUT "Limite DROITE fenetre (313) ";F2
340 INPUT "Limite HAUTE fenetre (6) "; F3
350 INPUT "Limite BASSE fenetre (243) "; F4
360 INPUT "Cadre desire (o/N) "; NT$
370 IF (NT$="N") OR (NT$="n") THEN 410
380 VG=F1-3: VD=F2+3: VH=F3-5: VB=F4+3
390 INPUT "Couleur du cadre (0 ... 7) "; CC: HCOLOR CC
400 IF (VG>0) AND (VD<MX-1) AND (VH>0) AND (VB<MY-1) THEN HPLOT VG,VH TO VD,VH: HPLOT VD,VH TO V
D,VB: HPLOT VG,VB TO VD,VB: HPLOT VG,VH TO VG,VB ELSE PRINT "Impossible ! ...": END
410 IF (TT$="E") OR (TT$="e") THEN ECH=F3: F3=MY-1-F4: F4=MY-1-ECH
```

Attention à rester à l'intérieur de l'écran !

Programme 5.1 ▶▶▶

▶▶▶

Beaucoup de renseignements à donner, pour plus de souplesse.

```
420 INPUT "Couleur du trace (0 ... 7) "; CT
430 INPUT "Bornes x,x' de la fonction (-2,2) "; X1,X2
440 INPUT "Bornes y,y' de la fonction (-2,2) "; Y1,Y2
450 INPUT "Nombre de lignes a tracer (7-19-23-27-31-43) "; NL
460 INPUT "Nombre de points par ligne (7-19-23-27-31-43) "; NP
470 INPUT "Distance de l'observateur RHO (8) "; RHO
475 INPUT "Distance de l'ecran (1) "; D
480 INPUT "Angle THETA en degres (60) "; THETA
490 INPUT "Angle PHI en degres (40) "; PHI
500 '
```

P0 comporte beaucoup de chiffres après la virgule, en prévision des machines haut de gamme à venir...

```
510 '...............  INITIALISATIONS  ...................
520 GOSUB 120: HV=1E25
530 DIM MIN(MX), MAX(MX)
540 FOR I= 1 TO MX: MIN(I)= HV: MAX(I)= -HV: NEXT
550 P0= 3.141592653/180: CY= MY-2: THETA= THETA*P0: PHI= PHI*P0
560 W1= SIN(THETA): M1= COS(THETA): W2= SIN(PHI): M2= COS(PHI)
570 W3= M1*W2: W4= W1*W2: W5= M1*M2: W6= W1*M2
580 '
590 X3= (X2-X1)/NP: Y3= (Y2-Y1)/NL
600 IF (THETA < 1) OR (THETA >179) THEN ECH= X1: X1= X2: X2= ECH: ECH= Y1: Y1= Y2: Y2= ECH: X3=
-X3: Y3= -Y3
610 L1= HV: L2= -HV: L3= HV: L4= -HV 'Surface de vision
620 FOR U= 0 TO NL
630   Y= Y2-U*Y3
640   FOR F= 0 TO NP
650     X= X1+F*X3
660     GOSUB 240
670     IF XA < L1 THEN L1= XA
680     IF XA > L2 THEN L2= XA
690     IF YA < L3 THEN L3= YA
700     IF YA > L4 THEN L4= YA
710   NEXT F
720 NEXT U
730 GOSUB 140
740 HCOLOR CT
750 U1= (F2-F1)/(L2-L1): U2= (F4-F3)/(L4-L3)
760 IF U2 < U1 THEN U1= U2
770 U2= U1
780 FOR U= 0 TO NL
790   Y= Y2-U*Y3: X= X1
800   GOSUB 240
810   E1= (XA-L1)*U1+F1: H1= CY-(YA-L3)*U2-F3
820   P= 2
830   IF MAX(E1) = -HV THEN 870
840   P= 0
850   IF MAX(E1) < H1 THEN P= 2
860   IF MIN(E1) > H1 THEN P= 1
```

HCOLOR choisit une couleur. A adapter selon le matériel possédé.

C'est ici que les calculs prennent du temps...

Programme 5.1 ▶▶▶

Les lignes 520 à 570 effectuent différentes initialisations. Les postes des tables de lignes de crêtes sont initialisés avec une valeur très élevée (1E25). Les différents *sinus* et *cosinus* de **THETA** et **PHI** sont calculés une fois pour toutes, afin d'augmenter la rapidité d'exécution du programme.

▶▶▶

```
870    FOR F= 0 TO NP
880      X= X1+F*X3
890      GOSUB 240
900      E2= (XA-L1)*U1+F1: H2= CY-(YA-L3)*U2-F3
910      Q= P
920      IF MAX(E2) = -HV THEN GOSUB 1250: GOTO 1200
930      EG= 0: Q= 0
940      IF MAX(E2) < H2 THEN Q= 2
950      IF MIN(E2) > H2 THEN Q= 1
960      IF (P = 0) AND (Q = 0) THEN 1200
970      IF (P = 0) OR (Q = 0) THEN TB= P+Q: GOTO 1010
980      IF P = Q THEN GOSUB 1250: GOTO 1200
990      EG= 1: TB= 1
1000      IF P <> 1 THEN TB= 2
1010      ON TB GOTO 1020,1070
1020      JE= MIN(E2)-MIN(E1)-H2+H1
1030      IF JE = 0 THEN GOSUB 1250: GOTO 1200
1040      E3= (H1*E2-H2*E1-MIN(E1)*E2+MIN(E2)*E1)/JE
1050      H3= (H1*MIN(E2)-H2*MIN(E1))/JE
1060      GOTO 1110
1070      JE= MAX(E2)-MAX(E1)-H2+H1
1080      IF JE = 0 THEN GOSUB 1250: GOTO 1200
1090      E3= (H1*E2-H2*E1-MAX(E1)*E2+MAX(E2)*E1)/JE
1100      H3= (H1*MAX(E2)-H2*MAX(E1))/JE
1110      IF P = 0 THEN E1= E3: H1= H3: GOSUB 1250: GOTO 1200
1120      E4= E2: H4= H2: E2= E3: H2= H3
1130      GOSUB 1250
1140      E2= E4: H2= H4
1150      IF EG = 0 THEN 1200
1160     EG= 0: P= 0
1170     IF TB = 1 THEN TB= 2: GOTO 1010
1180     TB= 1
1190     GOTO 1010
1200     P= Q: E1= E2: H1= H2
1210   NEXT F
1220 NEXT U
1230 END
1240 '
1250 HPLOT E1,H1 TO E2,H2
1260   I1= INT(E1+.5): I2= INT(E2+.5)
1270   IF I1 <> I2 THEN 1320
1280   IF H2 < MIN(I2) THEN MIN(I2)= H2
1290   IF H2 > MAX(I2) THEN MAX(I2)= H2
1300 RETURN
1310 '
1320 FOR K= I2+1 TO I1
1330      R= (K*(H2-H1)+E2*H1-E1*H2)/(E2-E1)
1340      IF R < MIN(K) THEN MIN(K)= R
1350      IF R > MAX(K) THEN MAX(K)= R
1360   NEXT K
1370 RETURN
1380 '======================================================
1390 '======================================================
```

Les parenthèses sont optionnelles, elles augmentent la lisibilité.

Si TB = 1, ligne 1020.
Si TB = 2, ligne 1070.

C'est terminé !

HPLOT trace une droite.
A adapter selon le langage utilisé.

Fin du programme 5.1

A la ligne 590, les incréments **X3** et **Y3** selon l'axe des **X** et des **Y**, sont calculés en fonction des bornes de la zone rectangulaire à étudier. Lorsque **THETA** est compris entre 180 et 360 degrés, les bornes de l'intervalle d'étude sont permutées en ligne 600. Ceci dans le but d'effectuer un balayage toujours dans le même sens.

Surface de vision

La ligne 610 initialise la *surface de vision* à une valeur très élevée (HV = 1E25). Puis, des lignes 620 à 720, deux boucles imbriquées *(Y,* puis *X)* parcourent l'intervalle d'étude par pas *(X3, Y3),* en vue de rechercher la surface de vision en projection. A chaque fois, la fonction définie en ligne 240 est appelée.

Le programme « réfléchira » d'autant plus longuement avant de commencer son tracé, que les nombres de lignes à tracer et de points par ligne seront élevés. C'est l'une des parties du programme qu'il serait intéressant d'optimiser par l'emploi de sous-routines en assembleur, appelées depuis le BASIC. En ligne 730, la visualisation passe en mode *haute résolution* et le tracé peut dès à présent commencer, après calcul des unités d'échelles à la ligne 750.

Lignes cachées

Les calculs relatifs à la détermination des *lignes cachées, projections* et autres *manipulations graphiques,* sont effectués dans le bloc de lignes 780 à 1230, selon les algorithmes étudiés précédemment. C'est-à-dire à l'aide des codes **P** et **Q** ayant pour valeurs 0, 1 ou 2. Cette autre partie gagnerait de même à être optimisée par l'emploi de l'assembleur.

Coupage de crête

Plus particulièrement, les lignes 780 à 1000 effectuent les recherches sur **P** et **Q**, pour déterminer d'éventuelles intersections de lignes, et sous quelle forme elles se produisent. Le segment à tracer sera identifié par deux points, de coordonnées *E1,H1* de code **P** (ligne 810) et *E2,H2* de code **Q** (ligne 900). A la ligne 1010, si *TB=1,* ceci signifie que c'est la crête inférieure qui est coupée. Donc des lignes 1020 à 1050, c'est la table **MIN()** qui est utilisée, puis le point d'intersection *E3,H3* est calculé. Par contre, lorsque *TB=2,* c'est la crête supérieure qui est franchie. La table **MAX()** sera alors concernée, des lignes 1070 à 1100 et *E3,H3* calculé de la même façon. Aux lignes 1030 et 1080, *JE = 0* indique que le segment à tracer est parallèle ou confondu à la ligne de crête. Le tracé sera donc effectué.

A la ligne 1120, la coordonnée *E2,H2* du segment côté invisible est sauvegardée dans *E4,H4,* puis le tracé jusqu'au point d'intersection est préparé. La ligne 1130 provoque le tracé proprement dit. Les valeurs du segment côté invisible sont reprises en ligne 1140. A la ligne 1200, les coordonnées et le code du dernier point calculé deviendront ceux du point précédent, pour le calcul suivant. Ceci dans le cas où *P=0* et *Q=0,* c'est-à-dire lorsque le segment est complètement caché. Rappelons les différents cas possibles de **P** et **Q**:

Cas possibles

- *P=2* et *Q=2* ou *P=1* et *Q=1*: le segment est visible.
- *P=2* et *Q=0* ou *P=1* et *Q=0* ou *P=0* et *Q=1* ou *P=0* et *Q=2*: le segment est partiellement caché et coupe l'une des lignes de crêtes.
- *P=0* et *Q=0*: le segment est complètement caché.
- *P=1* et *Q=2* ou *P=2* et *Q=1*: les deux lignes de crêtes sont coupées ensemble par le segment et la variable *EG* permet un aiguillage pour le calcul de la deuxième intersection, après celui de la première.

Enfin des lignes 1250 à 1300, le tracé est réellement effectué, par portions de droite. Ensuite les tables de lignes de crêtes sont mises à jour par interpolation linéaire. A partir des abscisses du tracé, *I1* et *I2* sont arrondis. Puis le calcul de l'ordonnée correspondante est effectué, des lignes 1320 à 1370.

Le tracé des fonctions en trois dimensions est maintenant terminé. Il est expressément recommandé d'essayer ce programme sur un grand nombre de cas, afin de bien assimiler les mécanismes et les influences des différents choix possibles.

2 LA REPRESENTATION D'OBJETS

Les calculs pour la représentation d'objets en trois dimensions seront mis en œuvre par le programme 5.2, écrit de façon identique au logiciel 5.1. Tout ce qui a été dit ci-dessus reste vrai. L'accent sera donc mis essentiellement sur ce qui différera fondamentalement du précédent.

Ecran ou table traçante

Dès que le programme est lancé, il est demandé à l'utilisateur si le tracé devra être effectué sur un *écran* ou une *table traçante.* En effet, dans le premier cas l'origine des axes se trouve en haut à gauche, alors que dans le second c'est la représentation mathématique qui est retenue (origine en bas à gauche). Il est donc important d'en tenir compte pour la suite des calculs. Puis les limites éventuelles d'une *fenêtre* sont demandées. Ainsi tout tracé devra être inscrit dans cette fenêtre, ce qui impliquera un *découpage* en cas de débordement et un arrêt du tracé hors des limites fixées. En cas d'erreur, le programme ne permet pas de dépasser la surface totale de l'écran. Il est possible de tracer un cadre de couleur déterminée autour du dessin. Mais contrairement au programme de tracé de fonctions, ici le cadre doit être rigoureusement tangent aux limites fixées; il s'agit en fait d'une ouverture par laquelle l'utilisateur voit un objet et non pas une simple mise en valeur artistique. La couleur du tracé est aussi demandée, ce qui aura son importance au niveau du relief.

Découpage en cas de fenêtre

Puis suivent les traditionnelles informations concernant la position de l'observateur dans l'espace, avec les mêmes recommandations que celles émises pour le programme 5.1. Les nombres figurant entre parenthèses sont ceux conseillés, lors d'un premier lancement par exemple. Enfin une dernière question demande si le programme doit tracer les détails internes à chaque face. En cas de réponse négative, seules les arêtes seront dessinées, ce qui est commode lors des premiers essais destinés à rechercher un cadrage rapide optimum. Le programme commence alors ses calculs, beaucoup plus rapidement que lors du tracé de fonctions.

Représentation réelle

Ce programme automatise de même entièrement la visualisation graphique. C'est-à-dire que l'objet sera dessiné selon une représentation **réelle** dans l'espace, en fonction du point de vue et en éliminant tout ce qui est normalement caché à l'observateur. La figure 5.3 a été obtenue en définissant successivement quatre fenêtres de taille identique sur l'écran. Le micro-ordinateur portable, répertorié dans les lignes de *DATA* du programme 5.2, a été tracé (en 5.3a), à l'aide des paramètres suivants:

RHO=300
D=500
THETA=45
PHI=15

Découpage

Ces paramètres sont identiques pour la fenêtre 5.3b, sauf en ce qui concerne la distance de l'observateur **RHO** (150). Le *découpage* a alors parfaitement rempli son rôle. En 5.3c, il avait été spécifié **THETA**=150 et **PHI**=−20, ce qui a provoqué cette vue de l'arrière et par en-dessous. Enfin le côté supérieur gauche de l'ordinateur a été représenté en 5.3d par **THETA**=−45 et **PHI**=45.

La figure 5.4 a été obtenue de la même façon par le programme, mais en substituant les *DATA* des lignes 5000 à la fin par celles constituant le programme 5.3. Un secrétaire a donc remplacé le micro-ordinateur, toujours selon les mêmes principes.

Volume de vision

Voici maintenant quelques explications sur le programme proprement dit, qui diffère sensiblement de celui utilisé pour le tracé de fonctions. Reportez vous à la section traitant de la théorie, afin de mieux en saisir les principes. La principale différence concerne l'algorithme déterminant ce qui est caché et ce qui ne doit pas être tracé, c'est-à-dire ce qui franchit éventuellement le *volume de vision* ou la *fenêtre* précisée. Il faut en effet admettre que cette fenêtre implique

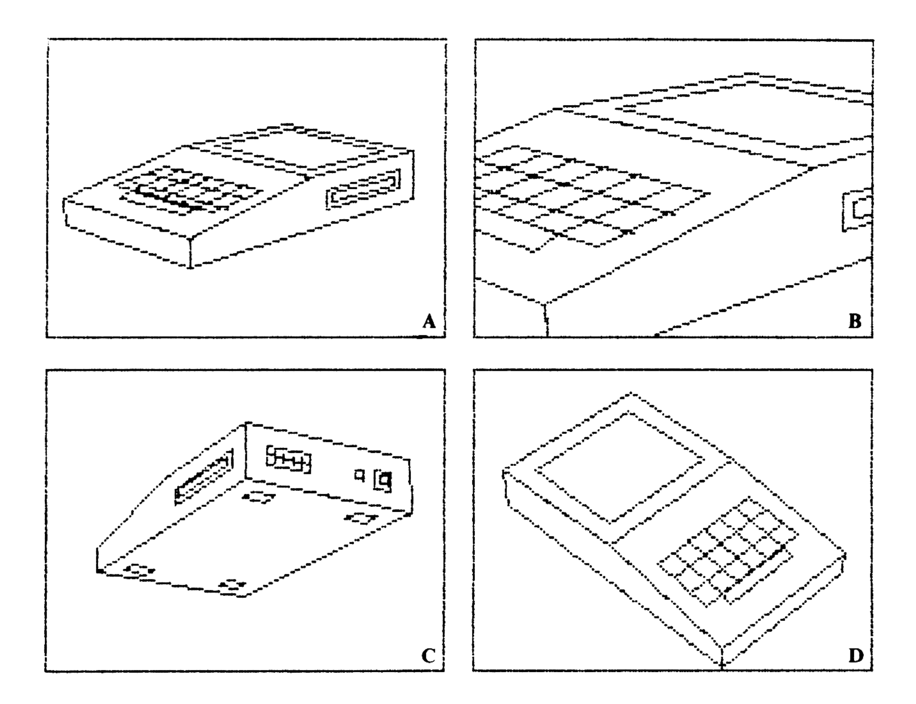


Figure 5.3 — Représentation réelle d'un micro-ordinateur, en trois dimensions, avec effacement automatique des surfaces cachées. Entre « A » et « B » seule la distance a été modifiée, en diminuant RHO, ce qui a introduit un découpage. En « C » et « D », le point de vue a été changé, par modification des angles THETA et PHI.

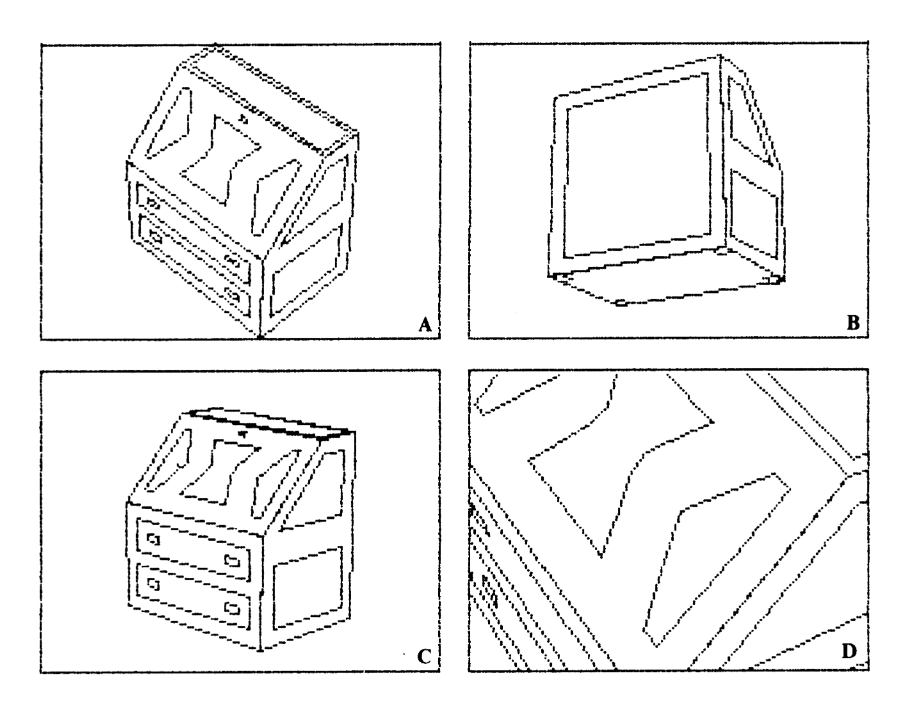


Figure 5.4 — Représentation d'un secrétaire en trois dimensions et vue réelle. En « A », « B » et « C », déplacement du point de vue par modification des angles THETA et PHI. En « D », rapprochement et découpage, par diminution de RHO.

momentanément une surface virtuellement rétrécie de l'écran. Ne pas définir de fenêtre se résume à utiliser une fenêtre égale aux dimensions maximales de l'écran.

Pyramide de vision

Afin de mieux saisir ce qui suit, reportez vous à la figure 4.7. Elle permet de déterminer une *pyramide de vision* dans l'espace, dont le sommet est situé à l'emplacement de l'observateur (Point de vision **O1,O2,O3** de la ligne 1280, ou **O'** de la figure 4.7) et les quatre plans s'appuyant sur les bords de la fenêtre. Ceci dans le système de vision défini par les axes **X1**, **Y1** et **Z1** (ligne 210) ou **XO**, **YO** et **ZO** de la figure. **Z1** (ou **ZO**) passe par le centre de la fenêtre, défini par **XM** et **YM**, tandis que **DX** et **EY** (ou **XE** et **YE**) en sont la demi-largeur et la demi-hauteur (lignes 1230-1240). Plusieurs algorithmes peuvent être employés.

Algorithme de COHEN et SUTHERLAND

L'un des plus performants est celui qui a été mis au point par Dan **COHEN** et Yvan **SUTHERLAND**. Il permet un découpage rapide, efficace et précis du tracé. La méthode consiste à donner un code binaire différent aux régions de l'espace situées autour du *volume de vision* et sa *fenêtre*. Pour en simplifier la démonstration, il est possible de l'appliquer au tracé en deux dimensions sur le plan. Au niveau de la troisième dimension, il suffira de remplacer la notion de surfaces planes par des volumes, le principe étant en tous points identique. Voici cette méthode de représentation, simplifiée, mais explicite:

1 0 0 1	1 0 0 0	1 0 1 0
0 0 0 1	0 0 0 0	0 0 1 0
0 1 0 1	0 1 0 0	0 1 1 0

Codage binaire

Elle montre, d'une part, que huit régions peuvent être définies autour de la fenêtre et que, d'autre part, à chacune de ces régions peut être affecté un code binaire sur quatre bits. Le rang de chaque bit s'étend de la droite vers la gauche. La convention suivante a été retenue:

- bit de rang 1=« 1 » : le point est situé **à gauche** de la fenêtre.
- bit de rang 2=« 1 » : le point est situé **à droite** de la fenêtre.
- bit de rang 3=« 1 » : le point est situé **en bas** de la fenêtre.
- bit de rang 4=« 1 » : le point est situé **en haut** de la fenêtre.
- sinon le bit aura pour valeur « 0 ».

Ceci permettra de coder un point situé *à gauche* par « **0001** », *à droite* par « **0010**, *en bas* par « **0100** » et *en haut* par « **1000** ». Donc le code correspondant à la fenêtre elle-même sera représenté par « **0000** » et par composition, les quatre autres régions situées dans les diagonales se verront attribuer les codes:

- Nord-Ouest: « **1001** ».
- Nord-Est: « **1010** ».
- Sud-Ouest: « **0101** ».
- Sud-Est: « **0110** ».

Chaîne de 4 bits

Il va donc être nécessaire de coder chaque point selon ce principe. Ceci est réalisé par le sous-programme s'étendant des lignes 360 à 420. En ligne 360 les quatre bits sont initialisés à « **0** » (*E1$*, *E2$*, *E3$*, *E4$*). *A1* et *A2* sont les coordonnées du point à tester. Ainsi lorsque *A1* sera inférieur à la limite gauche de la fenêtre, ceci signifiera que le point est situé à la gauche de celle-ci et le bit de rang **1** prendra la valeur: « 1 » (*E1$*); dans le cas contraire, il restera à « 0 ». Le même procédé est appliqué aux bits de rang **2** à **4** (*E2$* à *E4$*), selon les résultats des tests concernant la position du point en fonction de la fenêtre. A la ligne 410 la variable *A8$* représentera donc la chaîne de quatre bits, par concaténation de *E1$* à *E4$*. De ce fait, tout segment défini par les deux codes des coordonnées de ses extrémités, sera situé par rapport à la fenêtre selon trois possibilités:

```
10 '**********************************************************
20 '*****  OBJETS EN TROIS DIMENSIONS, SURFACES CACHEES  *****
30 '**********************************************************
40 '
50 GOTO 920
60 '
70 '              ================================
80 '============  bloc des commandes specifiques  ============
90 '              ================================
100 CLS: RETURN 'Effacement ecran Basse Resolution
110 MX= 320: MY= 250: RETURN 'Surface ecran H.R. (Nbe de points)
120 HDISP 1: RETURN 'Visu Basse Resolution
130 HDISP 2: RETURN 'Visu Haute Resolution
140 'HCOLOR x => Couleur du trace (0 a 7)
150 'HPLOT x,y => Trace un point
160 'HPLOT @ TO x',y' => Trace une droite, depuis dernier point
170 'HPLOT x,y TO x',y' => Trace une droite



180 '==========================================================
190 '
200 '*****  ALGORITHME DE Dan COHEN & Yvan SUTHERLAND  ******
210 X1= -X*ST+Y*TC: Y1= -X*CS-Y*SS+Z*CP: Z1= -X*ZZ-Y*SC-Z*SP+RHO
220 A1= X1: A2= Y1: A3= Z1
230 GOSUB 360
240 IF (A$ = "D") THEN 310
250 A7$= A8$: B4$= A8$: B4= X1: B5= Y1: B6= Z1
260 IF (A7$ <> "0000") THEN RETURN
270 A4= X1: A5= Y1: A6= Z1
280 A$= "M"
290 GOSUB 870
300 RETURN
310 B7$= A8$: B8$= A8$: B8= X1: C1= X1: C2= Y1: C3= Y1: C4= Z1: D1= Z1
320 GOSUB 440
330 A7$= B8$: B4$= B8$: B4= C1: B5= C3: B6= D1
340 RETURN
350 '
360 E1$= "0": E2$= "0": E3$= "0": E4$= "0"
370 IF (A1 < -DX/D*A3) THEN E1$= "1": GOTO 390
380 IF (A1 > DX/D*A3) THEN E2$= "1"
390 IF (A2 < -EY/D*A3) THEN E3$= "1": GOTO 410
400 IF (A2 > EY/D*A3) THEN E4$= "1"
410 A8$= E4$+E3$+E2$+E1$
420 RETURN
430 '
440 IF (A7$ = "0000") AND (B7$ = "0000") THEN 810
450 IM= 0
460 FOR JN= 1 TO 4
470   X$= MID$(A7$,JN,1): Y$= MID$(B7$,JN,1)
480   IM= IM+VAL(X$)*VAL(Y$)
490 NEXT JN
```

Bloc à adapter selon la machine possédée.

Merci à Dan et Yvan.

« 0000 » simule 4 bits.

« M » pour « MOVE », déplacement sans tracé.

Début des calculs binaires.

Programme 5.2 ▶▶▶

▶▶▶

```
500 IF (IM <> 0) THEN RETURN
510 C1$= A7$
520 IF (C1$ = "0000") THEN C1$= B7$
530 E1$= MID$(C1$,4,1)
540 IF (E1$ <> "1") THEN 580
550 T= (-DX/D*B6-B4)/(B8-B4+DX/D*(C4-B6))
560 A6= T*(C4-B6)+B6: A4= -DX/D*A6: A5= T*(C2-B5)+B5
570 GOTO 720
580 E2$= MID$(C1$,3,1)
590 IF (E2$ <> "1") THEN 630
600 T= (DX/D*B6-B4)/(B8-B4-DX/D*(C4-B6))
610 A6= T*(C4-B6)+B6: A4= DX/D*A6: A5= T*(C2-B5)+B5
620 GOTO 720
630 E3$= MID$(C1$,2,1)
640 IF (E3$ <> "1") THEN 680
650 T= (-EY/D*B6-B5)/(C2-B5+EY/D*(C4-B6))
660 A6= T*(C4-B6)+B6: A4= T*(B8-B4)+B4: A5= -EY/D*A6
670 GOTO 720
680 E4$= MID$(C1$,1,1)
690 IF (E4$ <> "1") THEN 720
700 T= (EY/D*B6-B5)/(C2-B5-EY/D*(C4-B6))
710 A6= T*(C4-B6)+B6: A4= T*(B8-B4)+B4: A5= EY/D*A6
720 IF (C1$ <> A7$) THEN 770
730 B4= A4: A1= A4; B5= A5: A2= A5: B6= A6: A3= A6
740 GOSUB 360
750 A7$= A8$
760 GOTO 440
770 B8= A4: A1= A4: C2= A5: A2= A5: C4= A6: A3= A6
780 GOSUB 360
790 B7$= A8$
800 GOTO 440
810 IF (B4$ <> "0000") THEN A4= B4: A5= B5: A6= B6: A$= "M": GOSUB 870
820 A4= B8: A5= C2: A6= C4
830 A$= "D"
840 GOSUB 870
850 RETURN
860 '
870 XE= D*A4/A6+XM: YE= YM-(D*A5/A6)
880 IF (A$ = "M") THEN IF (TT$="T") OR (TT$="t") THEN HPLOT XE,MY-YE: RETURN  ELSE HPLOT XE,YE:
RETURN
890 IF (TT$="T") OR (TT$="t") THEN HPLOT @ TO XE,MY-YE: RETURN  ELSE HPLOT @ TO XE,YE: RETURN
900 '
```

HPLOT trace un point dans ce cas. A adapter selon votre BASIC.

Le caractère « @ » indique que la droite à tracer débute au dernier point tracé. A adapter selon votre BASIC.

Le travail ne débute qu'ici !

Le sens du tracé peut en dépendre.

```
910 '****************** DEBUT DU PROGRAMME ******************
920 GOSUB 110: GOSUB 120: GOSUB 100 'Effacement ecran B.R.
930 INPUT "Type de visualisation : <E>cran ou <T>able tracante "; TT$
940 INPUT "Limite GAUCHE fenetre (6) "; F1
950 INPUT "Limite DROITE fenetre (313) "; F2
960 INPUT "Limite HAUTE fenetre (6) "; F3
970 INPUT "Limite BASSE fenetre (243) "; F4
```

Attention à rester dans les limites de l'écran !

Programme 5.2 ▶▶▶

▶▶▶

```
980 IF (F1 < 0) THEN F1= 0
990 IF (F2 > MX-1) THEN F2= MX-1
1000 IF (F3 < 0) THEN F3= 0
1010 IF (F4 > MY-1) THEN F4= MY-1
1020 INPUT "Cadre desire (o/N) "; NT$
1030 IF (NT$="N") OR (NT$="n") THEN 1070
1040 INPUT "Couleur du cadre "; CC: HCOLOR CC
1050 IF (TT$="T") OR (TT$="t") THEN HPLOT F1,MY-F3 TO F2,MY-F3: HPLOT F2,MY-F3 TO F2,MY-F4: HPLO
T F1,MY-F4 TO F2,MY-F4: HPLOT F1,MY-F3 TO F1,MY-F4: GOTO 1070
1060 HPLOT F1,F3 TO F2,F3: HPLOT F2,F3 TO F2,F4: HPLOT F1,F4 TO F2,F4: HPLOT F1,F3 TO F1,F4
1070 INPUT "Couleur du trace (0 ... 7) "; CT
1080 INPUT "Distance de l'observateur RHO (150) "; RHO
1090 INPUT "Distance de l'ecran (500) "; D
1100 INPUT "Angle THETA en degres (45) "; THETA
1110 INPUT "Angle PHI en degres (15) "; PHI
1120 INPUT "Y a-t-il des details (o/N) "; R$
1130 GOSUB 130: HCOLOR CT  'Init. Haute resolution
1140 GOSUB 1230  'Initialisations
1150 GOSUB 1330: GOSUB 1410  'Entree des coordonnees
1160 GOSUB 1690  'Vecteurs normaux
1170 IF (R$ = "o") OR (R$ = "O") THEN GOSUB 1530  'Entree des details
1180 GOSUB 1800  'Visibilite
1190 GOSUB 1980  'Trace des details
1200 END
1210 '
```

Beaucoup de renseignements confèrent plus de souplesse d'utilisation.

HCOLOR définit la couleur des tracés. A adapter selon le cas.

Perpendiculaires, pas ordinaires !

Programme 5.2 ▶▶▶

1) le segment est contenu entièrement **dans** la fenêtre: les deux codes sont donc égaux à « **0000** ».

2) le segment est situé entièrement **en dehors** de la fenêtre: les deux codes sont différents de « **0000** ».

3) **une seule extrémité** du segment se trouve à l'intérieur de la fenêtre: l'un des codes seulement sera égal à « **0000** ».

Test des bits

Le principe de l'élimination éventuelle d'un segment est donc très simple. Lorsque les deux codes sont égaux à « 0000 », ceci signifie que le segment est entièrement vu, donc il sera *tracé entièrement* (ligne 440). Sinon un calcul sera effectué sur ces deux codes, par application de la fonction « ET » *logique* (ligne 450 à 490). Si le résultat est différent de zéro, le segment est totalement invisible et ne sera donc *pas tracé* (ligne 500). Dans le cas contraire, une extrémité seulement se trouvera dans la fenêtre et un *découpage* sera effectué. C'est-à-dire que le programme devra rechercher les coordonnées du point d'intersection (unique ou double selon les cas) du segment avec les bords du cadre. Le tracé s'effectuera jusqu'à cette intersection seulement (ligne 510 à 800), avec retour au codage binaire et test de visibilité de la ligne 360, selon la complexité de la situation.

Move, déplacement

Draw, traçage

Des lignes 810 à 850 le traçage éventuel est préparé. Si le code binaire initial (sauvé en *B4$*) indique *non visible* (différent de « 0000 »), le déplacement s'effectuera sans traçage par l'action « M »ove contenue dans *A$*. Sinon, lorsque le code indique *visible* (égal à « 0000 »), le déplacement se fera avec traçage, par l'action « D »raw. Des lignes 870 à 890, l'action est effectivement menée, avec ou sans traçage (par tracé d'un simple point ou d'un segment complet), en tenant compte de l'emplacement de la fenêtre et du type de visualisation.

Angles en radians

Vecteurs normaux

Le reste du programme est classique. La ligne 1250 transforme en *radians* les angles **THETA** et **PHI**, préalablement entrés en *degrés*. Les lignes 1260 et 1270 effectuent des calculs répétitifs une fois pour toutes. Les coordonnées des différentes lignes à tracer seront contenues dans des lignes de *DATA* (lignes 5000 à la fin). Il ne reste plus aux lignes 1320 à 2110 qu'à entrer ces coordonnées, à calculer les *vecteurs normaux* des différentes faces selon ce qui a été étudié précédemment, à effectuer les tests de visibilité des différentes faces et enfin à en tracer éventuellement les détails.

▶▶▶

```
1220 '******************  INITIALISATIONS  *******************
1230 XM= INT((F2-F1+1)/2)+F1: YM= INT((F4-F3+1)/2)+F3
1240 DX= INT((F2-F1-2)/2): EY= INT((F4-F3-2)/2)
1250 THETA= THETA*3.14159265/180: PHI= PHI*3.14159265/180
1260 ST= SIN(TH): SP= SIN(PH): TC= COS(TH): CP= COS(PH)
1270 CS= TC*SP: SS= ST*SP: ZZ= TC*CP: SC= ST*CP
1280 O1=RHO*ZZ: O2=RHO*SC: O3=RHO*SP 'Point de vision
1290 HV=1E25  'Grande valeur
1300 RETURN


1310 '
1320 '***************  ENTREE DES COORDONNEES  ***************
1330 DIM S(50,3): 'Sommets (50 maximum)
1340 I= 0
1350 READ X,Y,Z
1360 IF (X = 1E25) THEN RETURN
1370 I= I+1
1380 S(I,1)= X: S(I,2)= Y: S(I,3)= Z
1390 GOTO 1350
1400 '
1410 DIM F(30,12) 'Faces (30 max., 12 sommets max. par face)
1420 I= 0 ' No. de face
1430 READ TS
1440 IF (TS = 0) THEN TF= I: RETURN
1450 I= I+1
1460 F(I,0)= TS+1
1470 FOR J= 1 TO TS
1480   READ F(I,J)
1490 NEXT J
1500 F(I,TS+1)= F(I,1)
1510 GOTO 1430
1520 '
1530 DIM PC(100,4), BC(2,20) 'Details
1540 I= 0: J= 1
1550 READ NB
1560 IF (NB <> 0) THEN 1600
1570 BC(1,I+1)= 0: BC(2,I+1)= J
1580 LD= I
1590 RETURN
1600 I= I+1
1610 BC(1,I)= NB: BC(2,I)= J
1620 READ X,Y,Z,C
1630 IF (C = -1) THEN 1550
1640 PC(J,1)= X: PC(J,2)= Y: PC(J,3)= Z: PC(J,4)= C
1650 J= J+1
1660 GOTO 1620
1670 '
```

Ces calculs sont effectués une fois pour toutes, pour gagner du temps.

C'est la taille mémoire qui introduit des limites, pas les capacités du programme !

Pour être sûr de comparer avec une valeur proche de l'infini...

Programme 5.2 ▶▶▶

▶▶▶

```
1680 '************  CALCUL DES VECTEURS NORMAUX  *************
1690 DIM N(50,3) ' 50 faces max. ===> 50 vect. norm. max.
1700 FOR I= 1 TO TF
1710   H1= F(I,1): H2= F(I,2): H3= F(I,3)
1720   T1= S(H2,1)-S(H1,1): T2= S(H2,2)-S(H1,2): T3= S(H2,3)-S(H1,3)
1730   S1= S(H3,1)-S(H1,1): S2= S(H3,2)-S(H1,2): S3= S(H3,3)-S(H1,3)
1740   U1= T2*S3-S2*T3: U2= T3*S1-S3*T1: U3= T1*S2-S1*T2
1750   N(I,1)= U1: N(I,2)= U2: N(I,3)= U3
1760 NEXT I
1770 RETURN
1780 '


1790 '****************  TEST DE VISIBILITE  ******************
1800 FOR I= 1 TO TF
1810   H1= F(I,1)
1820   B1= O1-S(H1,1): B2= O2-S(H1,2): B3= O3-S(H1,3)
1830   DF= B1*N(I,1)+B2*N(I,2)+B3*N(I,3)
1840   IF (DF <= 0) THEN 1940
1850   TS= F(I,0)
1860   N(I,0)= 1
1870   FOR J= 1 TO TS
1880     H1= F(I,J)
1890     X= S(H1,1): Y= S(H1,2): Z= S(H1,3)
1900     A$= "D"
1910     IF (J = 1) THEN A$= "M"
1920     GOSUB 210
1930   NEXT J
1940 NEXT I
1950 RETURN
1960 '


1970 '**************** TRACE DES DETAILS  ********************
1980 IF (R$ = "n") OR (R$ = "N") THEN RETURN
1990 FOR I= 1 TO TF
2000   IF (N(I,0) = 0) THEN 2100
2010   FOR J= 1 TO LD
2020     IF (BC(1,J) <> I) THEN 2090
2030     FOR L= BC(2,J) TO BC(2,J+1)-1
2040       X= PC(L,1): Y= PC(L,2): Z= PC(L,3): C= PC(L,4)
2050       A$= "D"
2060       IF (C = 0) THEN A$ = "M"
2070       GOSUB 210
2080     NEXT L
2090   NEXT J
2100 NEXT I
2110 RETURN
2120 '
2130 '
```

Pour ne tracer, par la suite, que ce qui doit être visible pour l'observateur.

« D » pour « DRAW », traçage...

... et « M » pour « MOVE », simple déplacement.

Programme 5.2 ▶▶▶

Papier millimétré

Etudions à présent la mise en œuvre du programme. Il convient tout d'abord de définir sur un *papier millimétré* l'objet à tracer, selon les conventions du dessin industriel. La figure 5.5 illustre cette technique. Différents tracés sont réunis

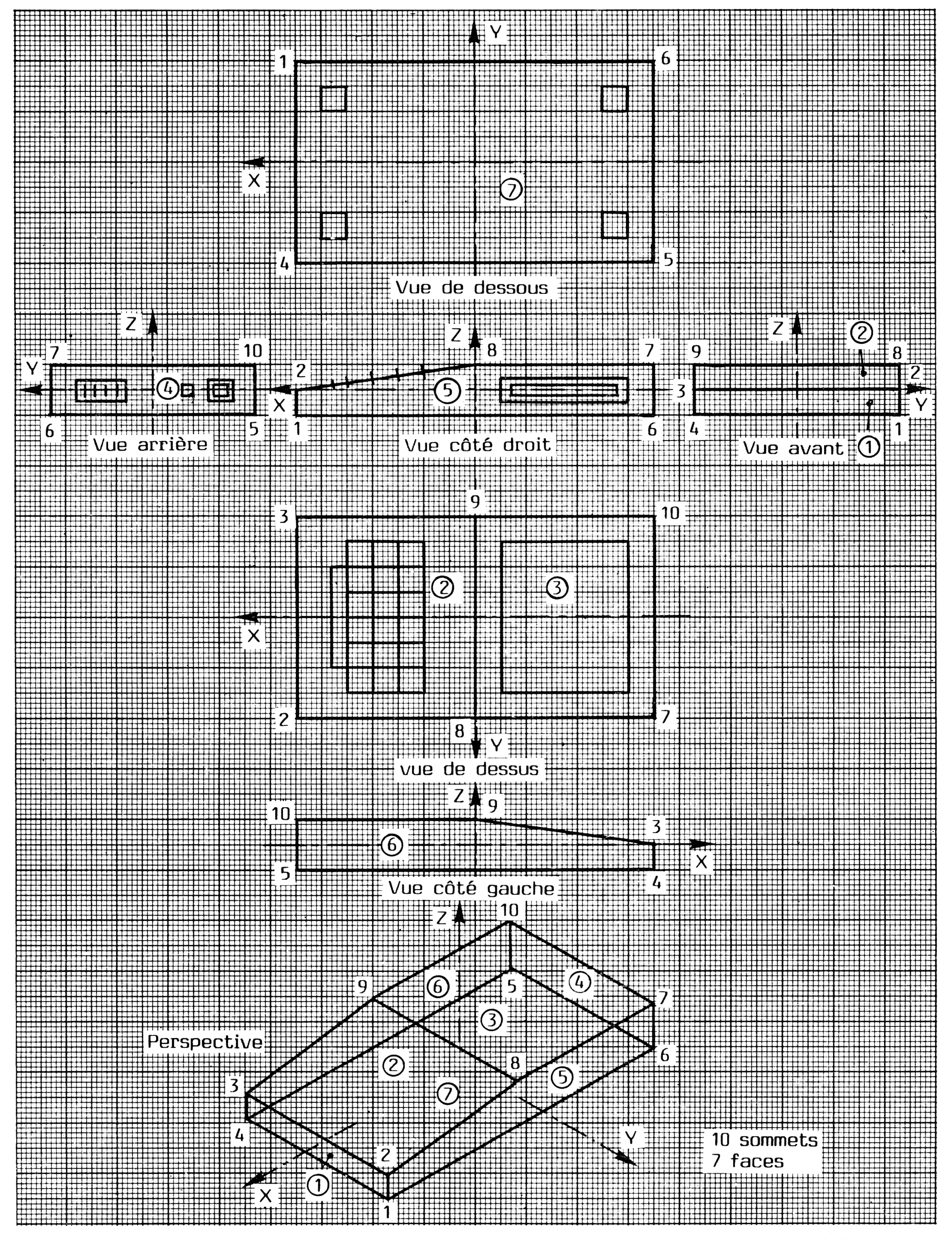


Figure 5.5 — Traçage de cotes du micro-ordinateur sur papier millimétré, sans préoccupation de la vision réelle.

▶▶▶

C'est à partir d'ici que l'on peut définir n'importe quel objet à tracer.

```
2140 '                        ##################
2150 '################### OBJET A TRACER #################
2160 '                        ##################
2170 '
5000 '##################### ORDINATEUR ####################
5010 '
5020 '=========
5030 ' SOMMETS  (Coord. x,y,z  ,  FIN: 1E25,0,0)
5040 '=========
5050 DATA 35,20,-6
5060 DATA 35,20,0
5070 DATA 35,-20,0
5080 DATA 35,-20,-6
5090 DATA -35,-20,-6
5100 DATA -35,20,-6
5110 DATA -35,20,6
5120 DATA 0,20,6
5130 DATA 0,-20,6
5140 DATA -35,-20,6
5150 DATA 1E25,0,0
5160 '
```

Indique la fin de la définition des emplacements des sommets.

```
5170 '=======
5180 ' FACES (Nbe Sommets, No Sommets sens trigo ext., FIN: 0)
5190 '=======
5200 DATA 4,1,2,3,4
5210 DATA 4,2,8,9,3
5220 DATA 4,8,7,10,9
5230 DATA 4,5,10,7,6
5240 DATA 5,6,7,8,2,1
5250 DATA 5,4,3,9,10,5
5260 DATA 4,4,5,6,1
5270 DATA 0
5280 '
```

« 0 » pour indiquer la fin de la série des faces, définies par les sommets qui les délimitent.

Les détails représentent des tracés éventuels, à l'intérieur de chaque face.

```
5290 '===========
5300 '  DETAILS (No Face / x,y,z , Action: 0 deplacement, 1 tracage)
5310 '=========== (Fin face: 0,0,0,-1  Fin generale: No de face= 0)
5320 '
5330 'DESSUS
5340 DATA 3
5350 DATA -5,-15,6,0
5360 DATA -5,15,6,1
5370 DATA -30,15,6,1
5380 DATA -30,-15,6,1
5390 DATA -5,-15,6,1
5400 DATA 0,0,0,-1
5410 '
5420 'DESSOUS
5430 DATA 7
5440 DATA 30,15,-6,0
5450 DATA 30,10,-6,1
5460 DATA 25,10,-6,1
5470 DATA 25,15,-6,1
5480 DATA 30,15,-6,1
5490 '
```

Indique la fin des détails d'une face.

Programme 5.2 ▶▶▶

▶▶▶

```
5500 DATA 30,-10,-6,0
5510 DATA 30,-15,-6,1
5520 DATA 25,-15,-6,1
5530 DATA 25,-10,-6,1
5540 DATA 30,-10,-6,1
5550 '
5560 DATA -25,-10,-6,0
5570 DATA -25,-15,-6,1
5580 DATA -30,-15,-6,1
5590 DATA -30,-10,-6,1
5600 DATA -25,-10,-6,1
5610 '
5620 DATA -25,15,-6,0
5630 DATA -25,10,-6,1
5640 DATA -30,10,-6,1
5650 DATA -30,15,-6,1
5660 DATA -25,15,-6,1
5670 DATA 0,0,0,-1
5680 '
5690 'DROITE
5700 DATA 5
5710 DATA -5,20,-2,0
5720 DATA -30,20,-2,1
5730 DATA -30,20,2,1
5740 DATA -5,20,2,1
5750 DATA -5,20,-2,1
5760 DATA -7,20,-1,0
5770 DATA -28,20,-1,1
5780 DATA -28,20,1,1
5790 DATA -7,20,1,1
5800 DATA -7,20,-1,1
5810 DATA 0,0,0,-1
5820 '
5830 'ARRIERE
5840 DATA 4
5850 DATA -35,15,-2,0
5860 DATA -35,5,-2,1
5870 DATA -35,5,2,1
5880 DATA -35,15,2,1
5890 DATA -35,15,-2,1
5900 DATA -35,13,-1,0
5910 DATA -35,13,1,1
5920 DATA -35,11,-1,0
5930 DATA -35,11,1,1
5940 DATA -35,9,-1,0
5950 DATA -35,9,1,1
5960 DATA -35,7,-1,0
5970 DATA -35,7,1,1
5980 DATA -35,15,0,0
5990 DATA -35,5,0,1
6000 '
```

```
6010 DATA -35,-6,-1,0
6020 DATA -35,-8,-1,1
6030 DATA -35,-8,1,1
6040 DATA -35,-6,1,1
6050 DATA -35,-6,-1,1
6060 '
6070 DATA -35,-11,-2,0
6080 DATA -35,-15,-2,1
6090 DATA -35,-15,2,1
6100 DATA -35,-11,2,1
6110 DATA -35,-11,-2,1
6120 DATA -35,-12,-1,0
6130 DATA -35,-14,-1,1
6140 DATA -35,-14,1,1
6150 DATA -35,-12,1,1
6160 DATA -35,-12,-1,1
6170 DATA 0,0,0,-1
6180 '
6190 'CLAVIER
6200 DATA 2
6210 DATA 25,-15,2,0
6220 DATA 25,15,2,1
6230 DATA 10,15,4,1
6240 DATA 10,-15,4,1
6250 DATA 25,-15,2,1
6260 '
6270 DATA 28,-10,1,0
6280 DATA 28,10,1,1
6290 DATA 25,10,2,1
6300 DATA 25,-10,2,1
6310 DATA 28,-10,1,1
6320 '
6330 DATA 25,-10,2,0
6340 DATA 10,-10,4,1
6350 DATA 25,-5,2,0
6360 DATA 10,-5,4,1
6370 DATA 25,0,2,0
6380 DATA 10,0,4,1
6390 DATA 25,5,2,0
6400 DATA 10,5,4,1
6410 DATA 25,10,2,0
6420 DATA 10,10,4,1
6430 DATA 20,-15,2,0
6440 DATA 20,15,2,1
6450 DATA 15,-15,3,0
6460 DATA 15,15,3,1
6470 DATA 0,0,0,-1
6480 '
6490 DATA 0
```

Fin générale des définitions des détails.

Fin du programme 5.2

sur cette feuille: la vue en **perspective** servant uniquement à repérer l'ordre des différents sommets et les vues de **dessus**, **dessous**, **droite**, **gauche**, **avant** et **arrière**. Les tracés peuvent s'effectuer à l'échelle, un millimètre représentant une unité de mesure. Les différents détails de chaque face doivent être tracés entièrement. Seuls deux impératifs sont à respecter: entrer les sommets de chaque face dans les lignes de *DATA* dans le sens *trigonométrique* vu de l'**extérieur** de la face; et surtout l'objet doit être de forme **convexe**. Ceci est dû à la méthode d'élimination des surfaces cachées utilisée, étudiée dans le chapitre relatif à la théorie. Nous verrons plus loin la façon de s'affranchir de cette contrainte, mais en étant obligé de visualiser toutes les surfaces. Il est important de bien orienter les faces selon les axes **X**, **Y** et **Z**, en notant soigneusement les numéros des sommets et des faces correspondants.

Sens trigonométrique

Lignes de DATA

Il ne reste plus qu'à reporter ces tracés dans les lignes de *DATA* du programme, qui se chargera automatiquement d'assembler et de visualiser les différentes faces d'une façon **réelle**. En premier lieu, il s'agit d'entrer les coordonnées **x,y,z** des sommets, selon leur numéro d'ordre. La fin de cette entrée sera indiquée par la coordonnée 1E25,0,0. (lignes 5050 à 5150). Puis les faces vont être définies (lignes 5200 à 5270).

La description de chaque face, dans l'ordre, comporte le nombre de sommets qui la compose, ainsi que les numéros de ces sommets, *impérativement* dans le sens *trigonométrique* vu de l'**extérieur** de la face. Un nombre de sommets égal à zéro, indique la fin de ces entrées. Il va falloir ensuite décrire éventuellement les détails composant chaque face. Il est nécessaire d'indiquer tout d'abord le numéro de la face concernée, puis de donner les coordonnées **x,y,z** de chaque point composant le tracé, avec à la fin l'action correspondante (**0** pour un déplacement *sans* tracé et **1** pour un déplacement *avec* tracé). La fin des détails de chaque face est indiquée par 0,0,0,−1. Ainsi, le début de tout tracé commence par une action « **0** » positionnant la « *plume* », suivi d'une série d'actions « **1** » effectuant le tracé « *plume baissée* ». En suivant les tracés sur le papier millimétré, il est aisé d'effectuer les reports correspondants. La fin générale est indiquée par un numéro de face égal à « **0** ».

0 : déplacement sans tracé
1 : déplacement avec tracé

La figure 5.6 permet de représenter un nouvel objet. Il s'agit d'un « secrétaire », dont les lignes de *DATA* correspondantes sont reprises dans le programme 5.3. Le résultat est montré par la figure 5.4.

Ces données sont destinées à remplacer celles du programme précédent, à partir de la ligne 5000 jusqu'à la fin.

```
5000 '******************** BUREAU *************************
5010 '
5020 '=========
5030 ' SOMMETS    (Coord. x,y,z  ,  FIN: 1E25,0,0)
5040 '=========
5050 DATA 20,30,-30
5060 DATA -20,30,-30
5070 DATA -20,-30,-30
5080 DATA 20,-30,-30
5090 DATA 20,-30,5
5100 DATA 20,30,5
5110 DATA -5,30,30
5120 DATA -5,-30,30
5130 DATA -20,-30,30
5140 DATA -20,30,30
5150 DATA 1E25,0,0
5160 '
```

Fin de la définition des sommets.

Programme 5.3 ▶▶▶

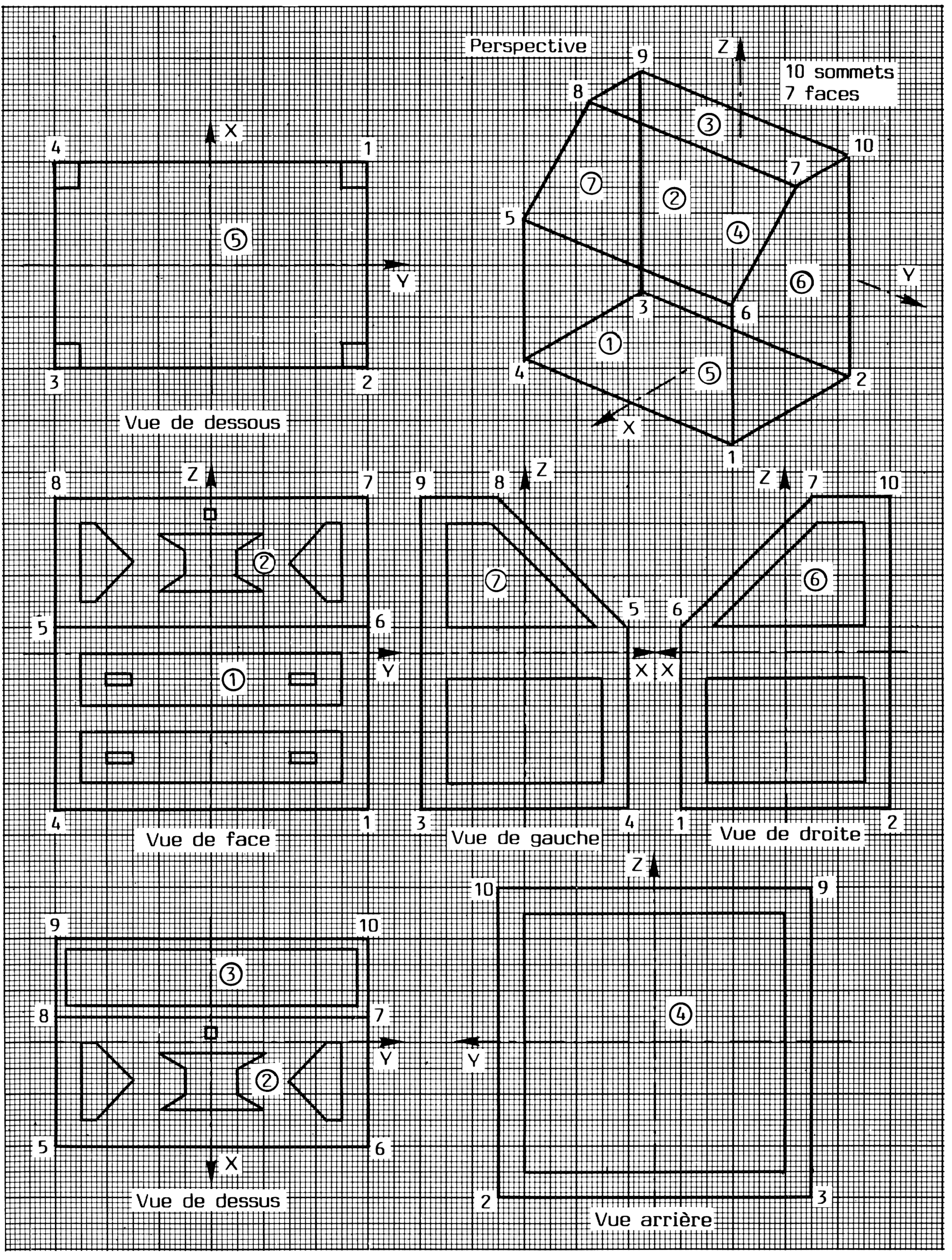


Figure 5.6 — Cotes, tracées sur papier millimétré, du secrétaire. Elles seront reportées dans les lignes de DATA du programme.

▶▶▶


```
5170 '=======
5180 ' FACES
5190 '=======
5200 DATA 4,5,4,1,6
5210 DATA 4,6,7,8,5
5220 DATA 4,7,10,9,8
5230 DATA 4,10,2,3,9
5240 DATA 4,2,1,4,3
5250 DATA 5,2,10,7,6,1
5260 DATA 5,3,4,5,8,9
5270 DATA 0
5280 '
5290 '=========
5300 ' DETAILS
5310 '=========
5320 '
5330 'DEVANT
5340 DATA 1
5350 DATA 20,-25,0,0
5360 DATA 20,25,0,1
5370 DATA 20,25,-10,1
5380 DATA 20,-25,-10,1
5390 DATA 20,-25,0,1
5400 DATA 20,-20,-4,0
5410 DATA 20,-15,-4,1
5420 DATA 20,-15,-6,1
5430 DATA 20,-20,-6,1
5440 DATA 20,-20,-4,1
5450 DATA 20,15,-4,0
5460 DATA 20,20,-4,1
5470 DATA 20,20,-6,1
5480 DATA 20,15,-6,1
5490 DATA 20,15,-4,1
5500 '
5510 DATA 20,-25,-15,0
5520 DATA 20,25,-15,1
5530 DATA 20,25,-25,1
5540 DATA 20,-25,-25,1
5550 DATA 20,-25,-15,1
5560 DATA 20,-20,-19,0
5570 DATA 20,-15,-19,1
5580 DATA 20,-15,-21,1
5590 DATA 20,-20,-21,1
5600 DATA 20,-20,-19,1
5610 DATA 20,15,-19,0
5620 DATA 20,20,-19,1
5630 DATA 20,20,-21,1
5640 DATA 20,15,-21,1
5650 DATA 20,15,-19,1
5660 DATA 0,0,0,-1
```



```
5670 '
5680 'COTE GAUCHE
5690 DATA 6
5700 DATA -7,30,25,0
5710 DATA -15,30,25,1
5720 DATA -15,30,5,1
5730 DATA 13,30,5,1
5740 DATA -7,30,25,1
5750 '
5760 DATA 15,30,-5,0
5770 DATA -15,30,-5,1
5780 DATA -15,30,-25,1
5790 DATA 15,30,-25,1
5800 DATA 15,30,-5,1
5810 DATA 0,0,0,-1
5820 '
5830 'COTE DROIT
5840 DATA 7
5850 DATA -7,-30,25,0
5860 DATA -15,-30,25,1
5870 DATA -15,-30,5,1
5880 DATA 13,-30,5,1
5890 DATA -7,-30,25,1
5900 '
5910 DATA 15,-30,-5,0
5920 DATA -15,-30,-5,1
5930 DATA -15,-30,-25,1
5940 DATA 15,-30,-25,1
5950 DATA 15,-30,-5,1
5960 DATA 0,0,0,-1
5970 '
5980 'DESSUS
5990 DATA 3
6000 DATA -18,-28,30,0
6010 DATA -18,28,30,1
6020 DATA -7,28,30,1
6030 DATA -7,-28,30,1
6040 DATA -18,-28,30,1
6050 DATA 0,0,0,-1
6060 '
6070 'DERRIERE
6080 DATA 4
6090 DATA -20,-25,25,0
6100 DATA -20,25,25,1
6110 DATA -20,25,-25,1
6120 DATA -20,-25,-25,1
6130 DATA -20,-25,25,1
6140 DATA 0,0,0,-1
6150 '
```


Programme 5.3 ▶▶▶

▶▶▶

```
6160 'DESSOUS
6170 DATA 5
6180 DATA -20,-30,-30,0
6190 DATA -20,-25,-30,1
6200 DATA -15,-25,-30,1
6210 DATA -15,-30,-30,1
6220 DATA -20,-30,-30,1
6230 '
6240 DATA -20,25,-30,0
6250 DATA -20,30,-30,1
6260 DATA -15,30,-30,1
6270 DATA -15,25,-30,1
6280 DATA -20,25,-30,1
6290 '
6300 DATA 15,25,-30,0
6310 DATA 15,30,-30.1
6320 DATA 20,30,-30,1
6330 DATA 20,25,-30,1
6340 DATA 15,25,-30,1
6350 '
6360 DATA 15,-30,-30,0
6370 DATA 15,-25,-30,1
6380 DATA 20,-25,-30,1
6390 DATA 20,-30,-30,1
6400 DATA 15,-30,-30,1
6410 DATA 0,0,0,-1
```

Fin des détails du dessous.

```
6420 '
6430 'ABATTANT
6440 DATA 2
6450 DATA -1,-25,26,0
6460 DATA -1,-22,26,1
6470 DATA 7,-15,18,1
6480 DATA 16,-22,9,1
6490 DATA 16,-25,9,1
6500 DATA -1,-25,26,1
6510 '
6520 DATA -1,22,26,0
6530 DATA -1,25,26,1
6540 DATA 16,25,9,1
6550 DATA 16,22,9,1
6560 DATA 7,15,18,1
6570 DATA -1,22,26,1
6580 '
6590 DATA 0,-10,25,0
6600 DATA 0,10,25,1
6610 DATA 6,5,19,1
6620 DATA 8,5,17,1
6630 DATA 15,10,10,1
6640 DATA 15,-10,10,1
6650 DATA 8,-5,17,1
6660 DATA 6,-5,19,1
6670 DATA 0,-10,25,1
6680 '
6690 DATA -2,-1,27,0
6700 DATA -2,1,27,1
6710 DATA -3,1,28,1
6720 DATA -3,-1,28,1
6730 DATA -2,-1,27,1
6740 DATA 0,0,0,-1
6750 '
6760 DATA 0
```

Fin des détails de l'abattant.

Fin générale de la définition des détails. A ne pas oublier !

Fin du programme 5.3

Objets convexes

Il ne faut pas hésiter à expérimenter largement, en visualisant toutes sortes d'objets, mais en veillant bien cependant à ce que ceux-ci soient de forme **convexe**. Il est tout de même très intéressant de n'avoir qu'à décrire les composantes d'un objet, en laissant le soin à un programme de le représenter sous n'importe quel angle, comme le ferait un professionnel du dessin.

CHAPITRE 6

LE MATERIEL

Avant d'utiliser avec profit les techniques étudiées jusqu'à présent, dans le but de les appliquer au **relief**, il est encore nécessaire de définir avec précision la nature du matériel qu'il sera nécessaire de réunir, en vue d'obtenir de superbes représentations . Ce matériel peut évidemment varier énormément, selon le type d'application désiré ou la plus ou moins grande qualité de reproduction recherchée. Le domaine d'exploration est assez varié, puisqu'il s'étend des lunettes *spéciales* jusqu'aux projections publiques, en passant par les procédés de reproduction photographique d'amateur ou professionnelle.

1. LA CONSTRUCTION DE LUNETTES

Effet stéréoscopique

Il s'agit du procédé de reproduction d'effet **stéréoscopique** le plus répandu à l'heure actuelle, au niveau de la photographie. Cette méthode ne permet d'obtenir que des vues en *monochromie*, mais avec une qualité de reproduction étonnante et surtout un prix de revient dérisoire. Récemment la télévision avait tenté un essai au niveau de la reproduction cinématographique en relief, mais n'avait pas obtenu le résultat escompté. Ceci était dû surtout au mode de retransmission des ondes hertziennes, qui atténuait fortement l'effet stéréoscopique. Pour se convaincre aisément du contraire, il suffisait de se procurer un exemplaire de la revue « Paris Match », numéro 1884 du 5 juillet 1985, qui non seulement proposait des photographies en relief d'excellente qualité, mais en plus offrait en prime une paire de lunettes spéciales.

Lunettes spéciales

Les lecteurs ayant eu la chance d'acquérir ce numéro, auront tout intérêt à conserver précieusement ces lunettes. Elles seront particulièrement utiles, lors de certaines de nos visualisations de fonctions ou d'objets en **relief**. Les autres devront les construire eux-même, ou les acquérir auprès d'autres revues, qui les proposent de temps en temps. Les Etablissements **3D VIDEO**, 91 rue du Faubourg Saint-Honoré, 78008 PARIS, proposent aussi ce genre de lunettes.

Afin d'apporter le maximum d'aide aux éventuels utilisateurs de ce procédé, voici une méthode simple, destinée à faciliter la construction de ce type de lunettes. Il est assez facile de réunir le matériel nécessaire. La *monture*, tout d'abord, peut être constituée de chutes de bristol ou autre carton du même genre. La figure 6.1 indique les dimensions de ses composantes. Les branches pourront être prévues ou non, selon le degré de simplicité recherché. Simple à réaliser, la construction nécessite seulement la reproduction sur papier millimétré des différents tracés. Il est même possible de ne se contenter que de reports approximatifs des cotes.

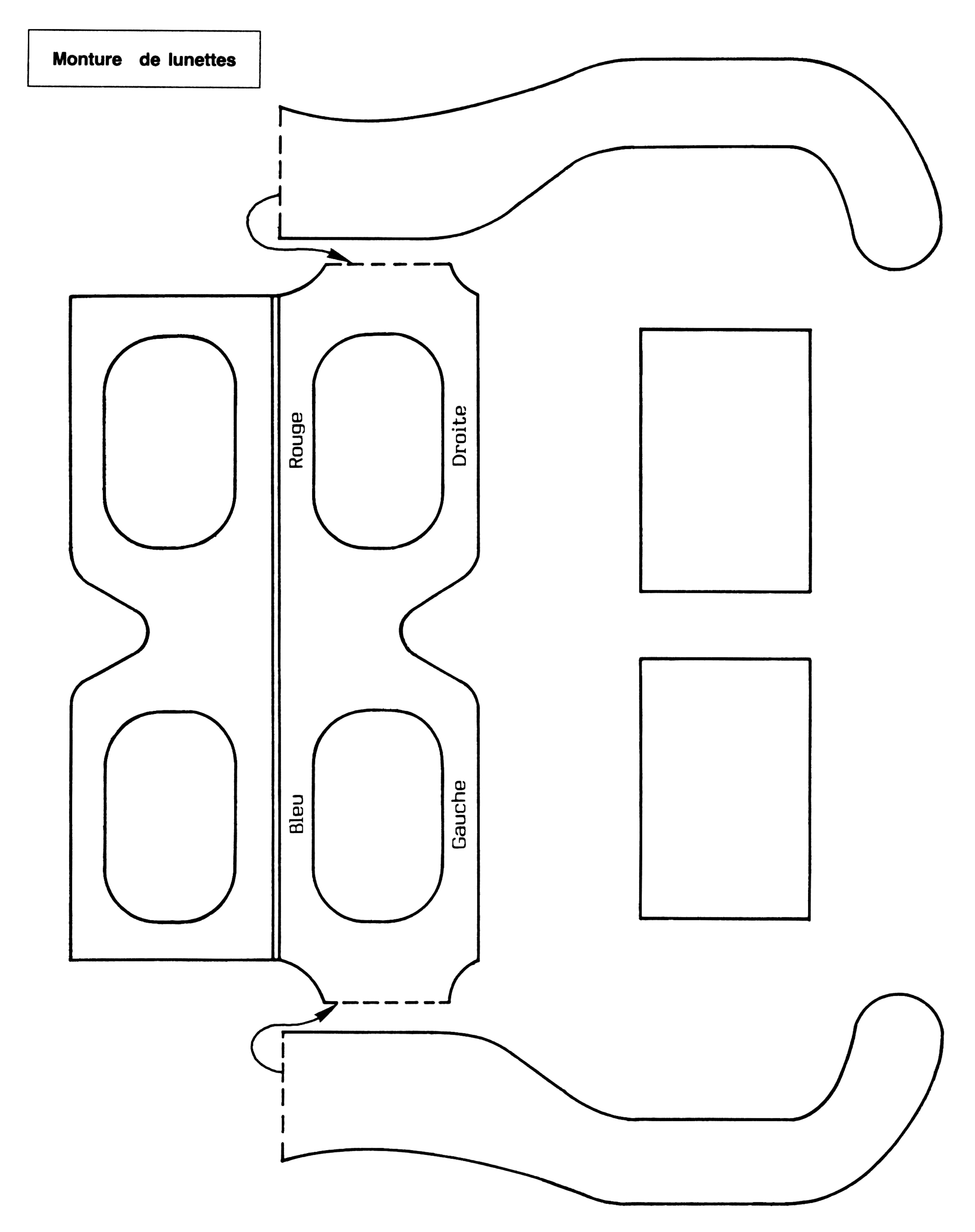


Figure 6.1 — Modèle de lunettes stéréoscopiques bicolores, en bristol, tracé à l'échelle 1.

Ensuite, l'utilisation d'un papier calque permettra de reporter les croquis sur le bristol. Puis, à l'aide d'un « cutter » ou d'une paire de ciseaux, les pourtours des tracés seront découpés soigneusement. La monture est alors prête à recevoir ses « verres ». A ce niveau, la difficulté peut être plus grande à surmonter, principalement lors de la recherche de ceux-ci. En effet, il est nécessaire de se procurer des feuilles de plastique de deux couleurs différentes: **rouge** et **bleu**. En fait, la couleur bleue peut varier d'un produit à l'autre et s'étendre du bleu profond au bleu-vert (*cyan*), en passant par le bleu clair. L'idéal est de rechercher la teinte se rapprochant le plus possible du type de bleu que l'écran du micro-ordinateur peut reproduire. En général un bleu moyen est disponible, ainsi que le *cyan*.

Verres rouges et verres bleus

D'autre part, il est possible de retoucher plus ou moins les réglages du moniteur ou du téléviseur, en vue d'obtenir le plus possible l'harmonisation des couleurs de l'écran et des verres de lunettes. Il existe un excellent moyen de rechercher les feuilles de plastique idéales: certains magasins revendeurs d'appareils photographiques offrent un choix de *filtres* dits « *gélatine* », servant en général aux trucages de toutes natures. Dans ce cas, le choix des teintes est assez vaste et ne devrait pas poser de difficultés insurmontables.

Filtres « gélatine »

Il est plus difficile de trouver d'autres matériaux satisfaisants, comme le verre ou le plastique rigide. De plus cette notion de teinte n'est pas très critique: des essais ont été effectués en changeant les couleurs des tracés du bleu au cyan, avec les mêmes lunettes et avec une parfaite qualité de reproduction *stéréoscopique*. Néanmoins, en vue de rechercher la perfection, il suffit de tracer des traits de différentes teintes sur l'écran du micro-ordinateur et de poser des échantillons de filtres sur ces derniers. Les teintes retenues pour le plastique des verres seront celles qui annuleront le plus possible la couleur *opposée*, tout en restituant au mieux la sienne.

Il suffit alors de découper ces verres selon le patron proposé, de les placer à l'intérieur des montures de carton, de rabattre les deux parties de ces dernières, l'une sur l'autre et de coller soigneusement le tout. Evitez tout débordement, préjudiciable à la qualité visuelle des filtres. Les lunettes sont terminées et prêtes à être expérimentées, dans le chapitre suivant.

Construction de lunettes stéréoscopiques

Si cette méthode de construction de lunettes *stéréoscopiques* est simple et efficace, elle n'en n'est pas moins destinée à un usage d'amateur. Pour une utilisation intensive et professionnelle, des montures plus rigides sont préférables. Il est possible, naturellement, d'acquérir de véritables montures de lunettes offrant toutes garanties de longévité pour un prix de revient intéressant, ou encore de construire soi-même ce type de montures, à base de corde à piano 8/10 à 15/10, disponible chez tous les revendeurs de matériel d'aéromodélisme. La figure 6.2 donne les cotes nécessaires, à respecter au moment des différentes torsions de ce « fil de fer », à l'aide de deux pinces à bec plat. Les « verres » seront ensuite collés sur ces montures, avec de la colle de bonne qualité (genre *cyanolite*).

2. L'HYPOTHESE D'UN STEREOSCOPE

Daguerréotype

Holographie

Le procédé de reproduction photographique en relief est relativement ancien. En effet, un an avant la naissance du *Daguéréotype*, un savant britanique (Charles **WHEATSTONE**) inventa le premier « stéréoscope » destiné à visualiser des dessins géométriques en relief. W.H.F **TALBOT** eut l'idée, par la suite, de visualiser avec cet appareil des images photographiques couplées. Or, fait extraordinaire, l'effet rendu avec ce type d'appareil a été parfait dès sa conception. Aucun autre procédé moderne n'a pu le supplanter, sauf en ce qui concerne le domaine de l'*holographie* (qui met en œuvre cependant des principes très différents).

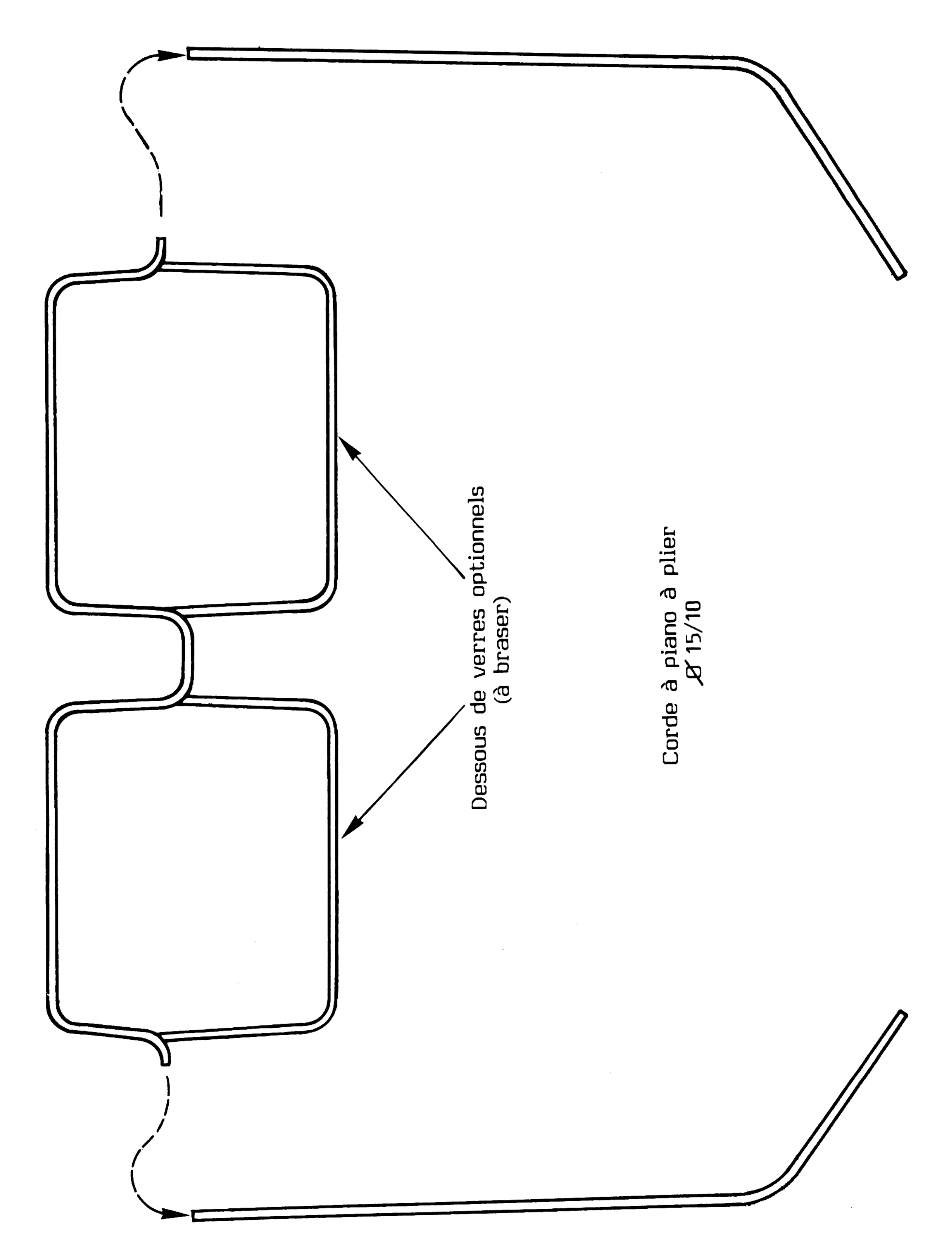


Figure 6.2 — Modèle, à l'échelle 1/1, de lunettes résistantes en cordes à piano, destinées à la projection publique avec filtres polarisants.

« Lestrade »

Cette méthode a connu un immense succès pendant toute la deuxième moitié du XIXe siècle, pour s'estomper après, sans jamais disparaître cependant. En effet, depuis de nombreuses années, des appareils de marque « Lestrade » offrent la même qualité de reproduction en relief, pour les différents sites touristiques français. Or, ce qui procure un intérêt primordial à ce procédé, c'est qu'il

Couleur, ET relief

est possible de nos jours de l'utiliser afin de visionner des vues en **couleur** et en **relief**. A l'aide de la technique de prise de vue précisée ci-avant, l'obtention de vues *stéréoscopiques* à partir de diapositives « 24 x 36 » est très facile: il suffit de « scotcher » deux vues correspondantes et de les glisser dans l'appareil. Le « Lestrade » est plus difficile à utiliser dans ce cas, tout en restant exploitable.

Stéréoscope

Plusieurs solutions s'offrent à l'amateur. La plus simple (et la plus onéreuse), consiste à acquérir un *stéréoscope* d'occasion. Seulement ce genre d'appareil est aujourd'hui assez rare. Il s'agit véritablement d'un objet de collection, concentré principalement sur des stands spécialisés, comme ceux de Bièvres ou Chelles.

Par contre le célèbre constructeur d'appareils photographiques, **PENTAX**, commercialise une visionneuse adaptée au relief, pour environ 470 F. Un autre constructeur offre par correspondance, pour 340 F, un véritable *stéréoscope*, offrant toutes les garanties de qualité: ce sont les **Ets Marcel ROYER** (Les Ricards, Plan de la Tour, 83120 Sainte maxime). Pour ce dernier appareil, tout a été prévu en vue d'obtenir les meilleurs résultats possibles, quel que soit l'utilisateur. Cet appareil permet d'utiliser des diapositives au format 24 x 36, sans montage particulier. Il est aussi possible d'utiliser deux visionneuses *monoculaires* couplées, genre **Agfa** ou **Gitzo**.

Hobbystirène

Une dernière possibilité est encore disponible, à peu de frais: construire un *stéréoscope* bon marché, en suivant le modèle décrit à la figure 6.3. Les oculaires sont disponibles chez les **Ets MULLER**, 17 rue des Plantes, 75014 PARIS, sous forme de deux loupes carrées de 60 mm de distance focale, pour 50 F environ. Le matériau utilisable peut être soit du contreplaqué de 5 mm d'épaisseur, soit du polystirène résistant (méthode dite du « HOBBYSTIRENE », qui a déjà fait ses preuves dans différents domaines). Le résultat obtenu sera très satisfaisant.

3. LA REPRODUCTION PHOTOGRAPHIQUE

Lorsque l'on désire visualiser des images en relief à partir de tracés sur micro-ordinateur, l'une des solutions envisageables consiste, entre autres, à photographier l'écran cathodique, ou employer les procédés modernes de reproduction photographique, présents sur le marché professionnel.

24×36 reflex

La première solution s'adresse surtout à l'amateur. La possession d'un simple appareil photographique « 24 x 36 » reflex est nécessaire. En effet, seul ce type d'appareil est susceptible d'annuler les erreurs de parallaxe dues à la visée. Il faut éviter des temps d'obturation trop courts (supérieurs au 1/30e), sous peine de reproduire la trame de balayage du tube cathodique. L'idéal est de disposer d'un trépied de caméra, très stable et robuste, ainsi que d'un *téléobjectif* (ou mieux un *zoom*), afin que la prise de vue soit suffisamment éloignée pour ne pas introduire de déformations. Réglez la luminosité de l'écran normalement, plutôt faiblement que trop poussée, afin d'éviter un certain « flou », au niveau des tracés. Il est préférable de travailler en lumière ambiante, atténuée. Si possible utiliser la prise de vue en pose, à l'aide d'un déclencheur souple. Les films de 100 ASA ou 21 DIN conviennent particulièrement à ce genre de prise de vue. Le réglage de base concerne un diaphragme réglé à « **f/8** » et un temps de pose de « **8** » secondes. En prenant au moins trois clichés identiques, mais avec des temps de pose réglés successivement à « 6 », « 8 » et « 10 » secondes, la qualité du résultat est pratiquement certaine.

PALETTE

Par contre, lors d'une utilisation professionnelle, une reproduction d'écran de haute qualité est indispensable. Elle peut être obtenue par le procédé « **PALETTE** », récemment mis au point pour l'**IBM-PC**. Ce produit professionnel, fabriqué par **Polaroïd**, est capable de réaliser des clichés de n'importe quel écran, de façon automatique. La qualité de reproduction est vraiment excellente, car tout un ensemble matériel et logiciel a été spécialement étudié en vue d'obtenir un maximum de qualité, dans le domaine de la recopie d'écran.

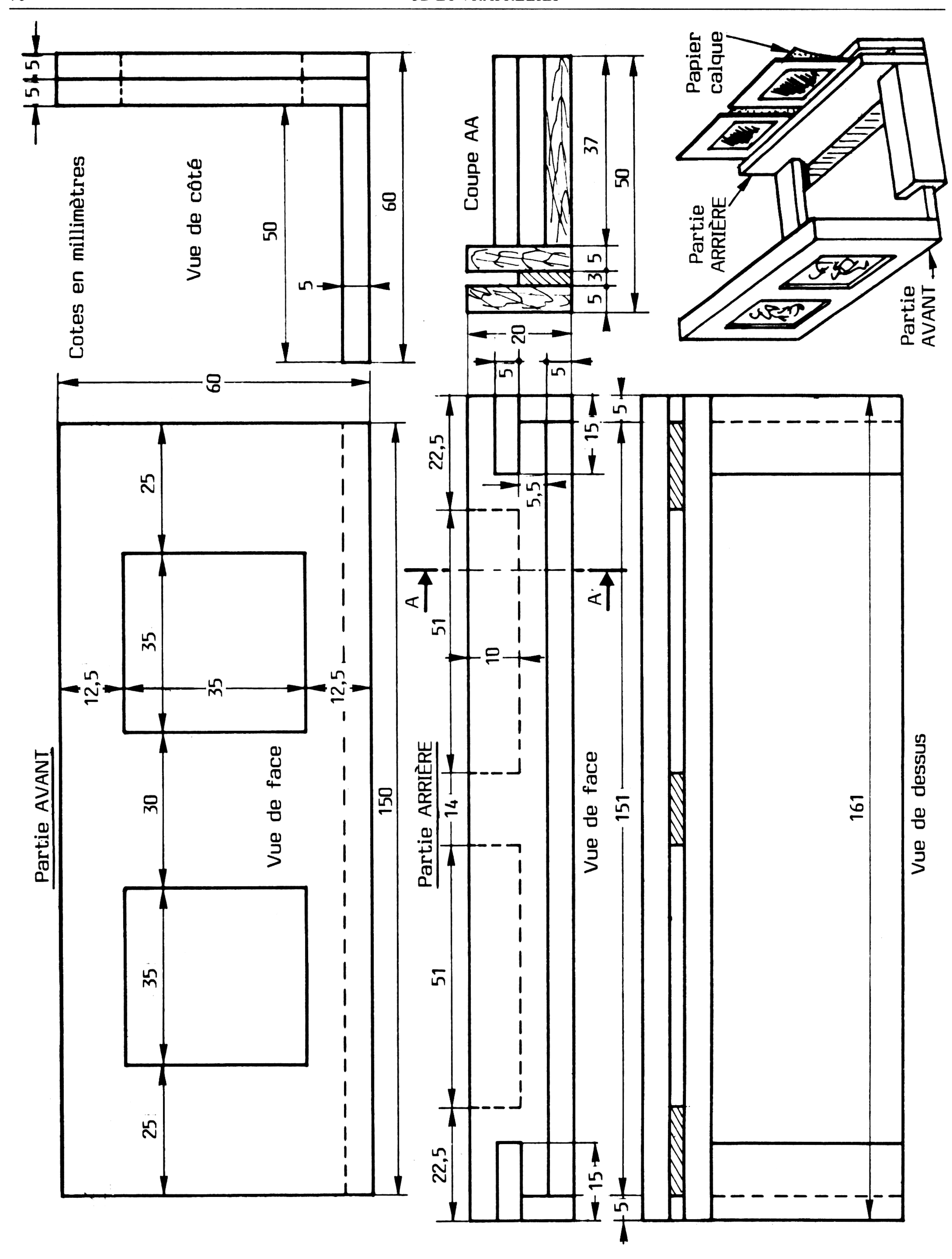


Figure 6.3 — Modèle de stéréoscope.
Sa réalisation pourra faire appel au « hobbystirène » ou au contreplaqué.

Si le prix de vente peut paraître un peu élevé (14 000 F), il est justifié par l'importance du matériel qu'il met en œuvre: une chambre noire recevant un boîtier d'appareil photo « reflex 24×36 » avec son moteur, un boîtier polaroïd et un câble de communication *asynchrone*, plus le logiciel gérant l'ensemble. Il s'agit en fait d'un véritable *périphérique*, branché sur la sortie vidéo de la carte *graphique/couleur*. La chambre noire envoie les images à l'appareil photo, sur asservissement du logiciel.

Asservissement logiciel

Sa mise en œuvre est très simple. Le logiciel modifie légèrement le **DOS**, de sorte qu'en utilisant les touches **[Schift]** + **[Prtsc]**, la recopie d'écran permet de stocker ce dernier dans un fichier, par l'intermédiaire d'un petit menu. Par la suite, le logiciel appelera ces différents fichiers au moment de les photographier. Les couleurs sont paramétrables, parmi une *palette* de 72 teintes.

En fait l'interactivité est très largement satisfaisante, puisqu'elle permet de visualiser les images avant de les photographier et de choisir le type de pellicule ou le nombre de vues désiré. Le procédé est entièrement automatisé, même au niveau des réglages des temps d'exposition en fonction du type de pellicule. Il est possible d'obtenir immédiatement soit des reproductions sur papier, grâce au boîtier polaroïd, soit des diapositives traitées avec le boîtier « 24 x 36 ». Dans le domaine des projections publiques, il s'agit très certainement de la solution idéale.

Polaroïd ou diapositives

4. LES PROJECTIONS PUBLIQUES

Ce domaine a déjà été abordé, au niveau de la reproduction photographique en *relief*. Il peut parfaitement être reproduit d'une façon similaire, pour les représentations d'objets ou de fonctions **tridimensionnelles**, issues d'un micro-ordinateur.

Projection en relief

Le principe reste identique à celui applicable aux lunettes bi-colores. Les spectateurs doivent impérativement recevoir deux images différentes, perceptibles par chaque œil d'une manière *simultanée*. Un procédé très efficace, qui de plus permet la reproduction, non seulement en relief mais en couleur, utilise des filtres polarisants. Le résultat est fantastique, aussi étudions ce procédé plus en détail.

Filtres polarisants

Il est tout d'abord nécessaire de réunir le matériel suivant:

- deux projecteurs de diapositives, de préférence avec télécommande électrique.
- un écran de projection *métallisé*.
- des filtres *polarisants* en bande plastique (grise).
- des lunettes à construire soi-même.

La figure 6.4 montre le fonctionnement de l'ensemble. La monture des lunettes peut directement s'inspirer de celle présentée à la figure 6.2. Le résultat obtenu est assez solide pour supporter de nombreuses manipulations de la part du public. Elles seront construites à partir de *corde à piano* 15/10. En ce qui concerne les « verres », ceux-ci seront fabriqués à l'aide de filtres spéciaux, dits *polarisants*, produits par Polaroïd et importés par la **Société SARELEC**, 86 avenue Jean-Jaurès, 91560 CROSNES (Tél. (16)69 49 15 66). Différents revendeurs existent en France, commercialisant ce produit: **Studio PERET**, 126 rue du Faubourg Saint-Martin, 75010 PARIS (Tél. (1) 42 06 96 91); **Photo Thyry**, 14 rue Saint Livier, 57000 METZ (Tél. (16) 87 62 52 19); **Photo Nobile**, 4 rue de l'Ange, 66000 PERPIGNAN (Tél. (16) 68 34 43 32).

Un filtre polarisant possède une propriété très importante pour le cas qui nous occupe: il ne laisse passer la lumière que dans *un seul sens*. Par exemple, lorsque l'on fait pivoter deux de ces filtres, superposés, il arrivera un moment où leur ensemble ne laissera plus passer la lumière. Lorsqu'ils sont positionnés perpendiculairement, ils paraissent *noirs*. Le procédé permettant d'exploiter cette particularité est simple à mettre en œuvre.

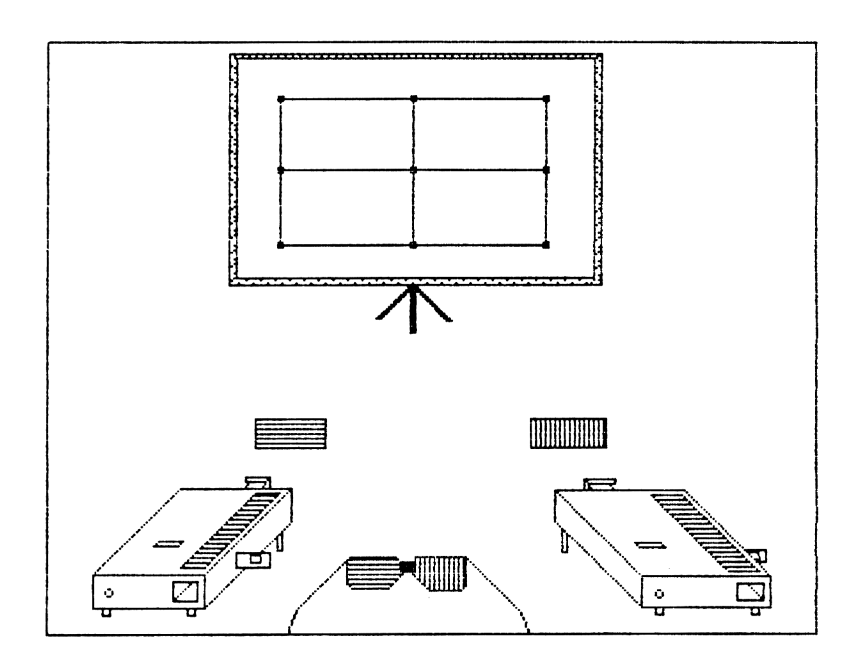


Figure 6.4 — Exemple de mise en œuvre d'une projection publique en relief, avec l'aide de deux projecteurs, de lunettes et de filtres polarisants.

Les deux projecteurs sont placés côte à côte et reçoivent chacun les vues qui lui sont propres: celui de gauche celles correspondant à l'œil gauche et celui de droite celles de l'œil droit. Seulement il faut intercaler un filtre *polarisant* sur le chemin des rayons issus des diapositives. Mais pas de n'importe quelle manière: les champs de polarisation des deux filtres doivent être perpendiculaires l'un par rapport à l'autre.

Ainsi la lumière provenant du projecteur de gauche sera *polarisée* dans le sens horizontal par exemple, tandis que celle émanant de celui de droite sera *polarisée* dans le sens vertical. Les deux images se superposeront sur l'écran, mais ne seront pas confondues. Afin que cette superposition soit parfaite, il est utile de régler, au préalable, les deux projecteurs. C'est-à-dire en employant deux diapositives identiques, dont les points de repère devront être confondus sur l'écran, en l'absence des filtres.

Ecran métallisé

D'autre part, il est important d'utiliser un écran *métallisé*, dont la surface ne doit pas être traitée. Un écran *perlé* provoquerait une dispersion de la lumière, préjudiciable à un bon rendu du relief. Par contre il est possible de métalliser un écran ordinaire, à l'aide d'une peinture métallisée. Les « verres » des lunettes seront aussi constitués de filtres *polarisants*, dont le sens de polarisation de chacun des verres devra *impérativement* être en relation avec celui du projecteur correspondant.

Effet de relief

Ce n'est que de cette façon que l'œil gauche des spectateurs ne recevra que l'image issue du projecteur de gauche et inversement pour l'œil droit. A l'aide de ce procédé, non seulement l'effet de **relief** est recréé, mais l'emploi de la couleur est rendu possible par le fait de la séparation totale des deux images, à la prise de vue, à la projection et enfin à la perception au travers des lunettes. Les spectateurs ne devront pas trop s'écarter de l'axe de l'écran, afin de profiter pleinement de l'effet recherché.

Maintenant toutes les connaissances nécessaires ont été réunies, pour qu'il soit enfin possible de mettre en œuvre le relief sur micro-ordinateur, d'une façon optimum. Il ne reste plus qu'à étudier les différentes modalités de création de cet effet.

CHAPITRE 7

LA MISE EN ŒUVRE

Maintenant que tous les éléments nécessaires aux dessins en trois dimensions d'une part et à leur exploitation d'autre part, sont supposés bien assimilés, plusieurs choix s'offrent à l'utilisateur au niveau de leur exploitation. Selon la sophistication du matériel informatique, ou les désidérata de chacun, les dessins peuvent être visualisés simplement en *trois dimensions*, préparés pour une *représentation ultérieure*, ou exploités instantanément, *en relief.* Mais il est aussi possible d'aller plus loin, en étudiant les effets d'animations réelles en trois dimensions, ainsi qu'un niveau récemment apparu dans le domaine des *courbes fractales*, c'est-à-dire leur extension à la troisième dimension.

1. LES DESSINS EN RELIEF

Les différentes catégories de dessins en **relief** varient selon les possibilités du micro-ordinateur possédé. Actuellement ceux-ci peuvent se subdiviser en deux catégories (puisque ceux ne disposant que de la basse résolution en *monochromie* appartiennent déjà au passé):

Une ou deux pages en haute résolution

— les machines permettant d'écrire ou de lire en **haute résolution couleur** sur **deux pages** différentes.

— celles n'offrant qu'**une page** en **haute résolution**, même en noir et blanc.

Dans ce dernier cas, la mise en œuvre des programmes est très simple. Après avoir choisi soit la représentation de **fonctions**, soit celle d'**objets**, un bon procédé consiste à effectuer quelques essais rapides, afin de déterminer la vue la plus satisfaisante. Puis, une fois les différents paramètres fixés d'une manière optimale, un dernier lancement, avec un maximum de finesse au niveau des tracés, génèrera le dessin définitif. Ce dernier sera sauvegardé selon trois possibilités différentes: soit par stockage sur une disquette si le logiciel le permet, soit par une sortie imprimante lorsque l'utilisateur en possède une ou bien encore en photographiant simplement l'écran.

Focale longue

Les meilleurs résultats sont obtenus par l'utilisation d'appareils photos « 24x36 », munis d'un objectif à *focale longue.* Cela dans le but d'éliminer les erreurs dues à la *parallaxe*, ainsi que d'éventuelles déformations. Le temps de pose doit être inférieur au 1/30ème de seconde, pour éviter des phénomènes de balayage parasites. Comme conseillé précédemment, il est préférable de travailler en pose de quelques secondes (8), pour une ouverture de f8 et à l'aide d'un film de 100 ASA. D'où l'intérêt d'utiliser un trépied stable et un déclencheur souple. Il est pratique d'utiliser des diapositives.

Diapositives

Stéréoscope

Comme on l'a vu, il est nécessaire de disposer de deux vues, décalées d'environ **5** à **6** degrés. Il faut donc noter soigneusement l'angle **THETA**, sélectionné au moment du premier dessin. La couleur du tracé choisie sera fonction de l'utilisation projetée. Si les vues sont destinées à être visualisées à l'aide d'un *stéréoscope*, ou par projection sur écran, la notion de couleur n'a guère d'importance et dépendra du goût de chacun. Par contre, dans le cas de l'utilisation avec les lunettes spéciales comportant les verres **bleus** et **rouges**, il est impératif de suivre la procédure suivante: Le premier tracé concernera la vue de gauche, donc devra être de couleur **bleue**. Quand au second tracé, il devra utiliser la couleur **rouge**. Dans tous les cas ce dernier doit être en tous points identique au premier, sauf en ce qui concerne la légère augmentation de l'angle *THETA*. Il constituera la vue de droite, stockée selon les mêmes procédés.

L'utilisateur pourra utiliser trois procédés de restitution différents, en fonction du type de stockage. L'une de ces possibilités concerne la reproduction des dessins sur imprimante. cette option est souvent utilisée en cas de possession d'un matériel minimum, car les résultats obtenus varient en fonction de la personne qui l'utilise. Le cerveau peut et doit être trompé, mais il existe des conditions plus favorables que d'autres, pour parvenir à ce résultat. Or certaines personnes sont plus réceptives que d'autres, au niveau des phénomènes de vision *stéréoscopiques*. Il ne faut par conséquent guère s'étonner du fait que tous les utilisateurs ne soient pas capables de percevoir l'effet de relief, à l'aide de cette technique simple.

A chaque œil son image

Elle consiste à effectuer une recopie d'écran de chaque vue. Puis à placer ces deux images l'une à côté de l'autre, sur un support plat. Comme l'œil droit ne doit percevoir que l'image de droite et l'œil gauche que celle de gauche, il est nécessaire de placer une feuille de carton de 30 cm de hauteur environ, entre les deux tracés. Après s'être positionné au-dessus du carton tenu verticalement et suite à un moment d'adaptation variable selon l'utilisateur, les deux images doivent sembler se fondre en une seule, lorsque le regard fixe la base du carton. D'ailleurs les personnes myopes possèdent même la faculté de se placer à quelques centimètres des dessins, tout en continuant à les voir nets. Dans ce cas le carton est inutile, chaque œil ne pouvant percevoir que la vue qui est située immédiatement en dessous de lui.

Fonction de synthèse

Avec ce procédé, les tracés doivent être suffisamment petits pour que leurs centres ne soient pas très éloignés l'un de l'autre. Le cerveau possède une puissante fonction de synthèse, mais elle a ses limites. Il est préférable de lui faciliter la tâche au maximum. En cas d'impossibilité à visualiser correctement le relief selon ce procédé, il est possible d'améliorer considérablement la fusion des deux vues à l'aide de tracés symétriques, tout en conservant fictivement un écart de six degrés environ entre les angles *THETA*. En intervenant au niveau de ces angles, ce résultat peut facilement être atteint. Ou bien effectuez les recopies d'écran sur du papier calque et retournez le dessin de droite par exemple. Avec un bon éclairage, ce dernier sera parfaitement perçu par transparence.

Par ce moyen, les deux images sont situées de part et d'autre d'un axe de symétrie imaginaire. A ce moment placez un petit miroir tenu verticalement, contre l'aile droite du nez, de façon à ce que l'œil gauche regarde la vue de gauche, tandis que l'œil droit ne puisse voir celle de droite que par réflexion sur la surface du miroir. Il est alors possible, en jouant sur l'inclinaison de ce dernier, de provoquer la superposition des deux images. Le relief sera alors parfaitement synthétisé, avec beaucoup plus de facilité, par le cerveau.

Vue en relief

Toujours en poursuivant dans cette voie, à l'aide de deux miroirs il n'est plus nécessaire d'obtenir des vues symétriques. Effectuez des essais, afin que les inversions s'annulent et puissent permettre la superposition des deux dessins. Enfin, avec un *stéréoscope* ancien, il suffit de placer les deux vues à l'intérieur de celui-ci, comme s'il s'agissait de diapositives. En orientant le miroir placé sur le dessus de l'appareil, l'intérieur du boîtier s'illuminera et la vue en *relief* deviendra possible.

Stéréoscope

Le deuxième procédé, un peu plus complexe à mettre en œuvre, nécessite la possession d'un *stéréoscope*, comme décrit plus avant. Les deux tracés seront effectués à l'aide de couleurs quelconques, pourvu que la seule différence, au niveau des paramètres sélectionnés, ne porte que sur l'angle *THETA* (Augmenté de six degrés pour la vue de droite, rappelons-le). Puis photographiez l'écran comme vu plus haut. Si le film employé génère des diapositives, montez les dans le *stéréoscope* telles quelles, ou scotchées ensemble, selon le modèle employé. La vision en relief sera parfaitement perçue. Il en sera de même dans le cas de tirage sur papier, à condition que le stéréoscope comporte le petit miroir d'éclairage, cité plus haut.

Projection publique en relief

Par contre il est très intéressant d'effectuer la reproduction photographique à l'aide de *diapositives*. Ce procédé permettra la projection publique *en relief*, comme étudié ci-avant. Avec les filtres polarisants placés de manière correcte devant les diapositives et les lunettes décrites, équipées elles aussi de ces mêmes filtres montés en conséquence, tout un public pourra s'émerveiller de la vision en relief, collectivement.

Relief sur moniteur

Double tracé

Mais le meilleur procédé demeure bien entendu celui permettant la vue en relief *directement sur l'écran*. Pour cela, les deux images doivent être tracées de deux couleurs différentes, celle correspondant à la vue de gauche étant tracée à l'aide de la couleur *cyan* (bleu-vert) et celle de la vue de droite de couleur rouge. La possession, ou la construction des lunettes décrites lors des chapitres précédents, est nécessaire. Elles devront comporter un « verre » de couleur *cyan* à gauche et *rouge* à droite. Si les tracés ne comportent pas trop de lignes s'effaçant mutuellement, ils pourront être produits sur le même écran, en respectant les couleurs précitées, mais aussi en augmentant l'angle *THETA* de la vue de droite, de six degrés, comme d'habitude.

Vision stéréoscopique

En regardant l'écran au travers des lunettes, la vision *stéréoscopique* est possible. Par contre, lorsque les images comportent un grand nombre de lignes, comme c'est le cas pour le tracé de fonctions par exemple, la solution idéale consiste à pouvoir écrire sur **deux pages** haute résolution différentes. Une page, sélectionnée en mode direct dans notre cas par la commande « **HPAGE 0** », comportera le tracé de la vue de gauche en bleu, tandis que « **HPAGE 1** » permettra de dessiner en rouge sur la deuxième page, la vue relative à l'œil droit. Il convient alors de provoquer un *clignotement* très rapide des deux pages et de visualiser cet effet à l'aide des lunettes. Pour ce faire il est possible d'utiliser en mode direct ou par programme, la séquence suivante :

FOR I=1 TO 50000: HPAGE 0: HPAGE 1: NEXT I.

Perception en relief

Le clignotement ne sera ressenti que comme un *scintillement*, mais la perception en **relief** sera absolument fantastique. Une image synthétisée sur un micro-ordinateur semblera alors « crever » l'écran et venir à la rencontre de son concepteur. L'effet de distance entre les différents plans est très nettement perceptible, comme pour les vues d'objets réels.

Par un stockage sur disquette de différentes images, il sera alors possible de visualiser sans délais les dessins désirés. Or de plus en plus de nouveaux micro-ordinateurs disposent de la possibilité de faire pivoter deux ou plusieurs pages, en haute résolution. C'est le cas de l'**Apple**, de la nouvelle norme **MSX II**, de l'*Amstrad CPC 6128*, de l'**Amiga** et même d'un ancêtre comme le **TRS 80**, ce dernier devant toutefois être équipé d'une carte *haute résolution couleur*. Pour les matériels ne possédant pas cette particularité, il serait évidemment possible de recréer cette possibilité artificiellement, au moyen de l'assembleur, par déplacements de zones de mémoire appropriées (Ce qui sortirait du cadre de cette étude).

Animation

Le lecteur désirant se procurer des filtres en *gélatine* obtiendra de bons résultats en acquérant des filtres « KODAK » numéro **38A** pour le *bleu* et **25** pour le *rouge*. Cependant, les possibilités des tracés en trois dimensions ne sont pas épuisées avec le relief. En retenant la possibilité de faire pivoter deux pages en haute résolution, l'**animation** devient possible, *en temps réel*.

2. LES ANIMATIONS

Les efforts entrepris en vue d'obtenir des tracés satisfaisants en **trois dimensions** et en **relief**, ont porté leurs fruits. Il serait dommage de s'arrêter en si bonne voie, alors que d'autres domaines d'application, du même ordre, restent encore à explorer. Nous allons maintenant étudier les possibilités d'**animation** des images obtenues en trois dimensions. Cela sans sacrifier notre objectif, qui concernait entre autre l'obtention de vues *réelles* par élimination des surfaces cachées.

Animation en temps réel

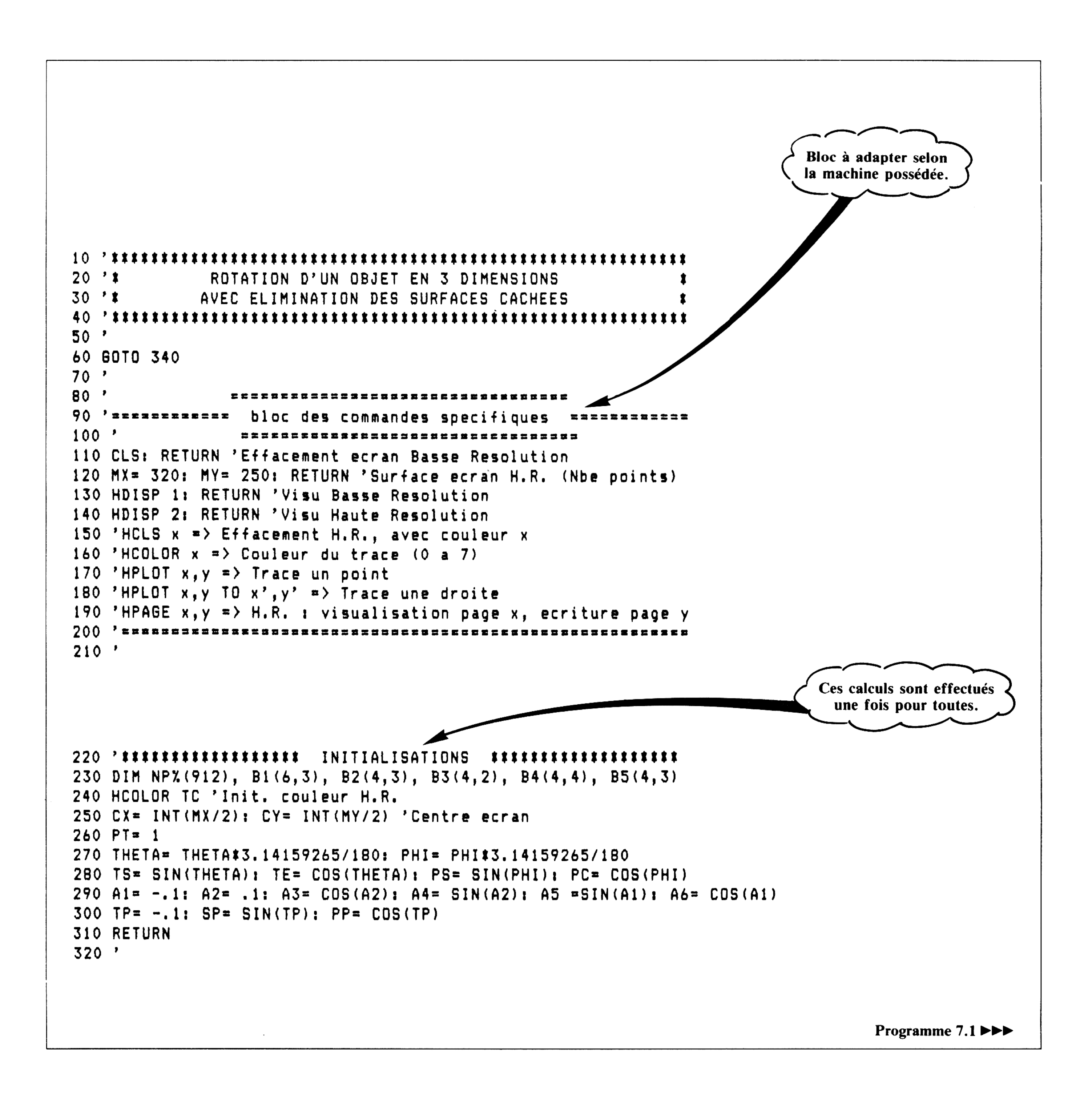
L'objet qui va se mouvoir sur l'écran représentera une pyramide. Ce sujet a été choisi dans le but de satisfaire à une double contrainte: obtenir une animation en temps réel suffisamment rapide pour être intéressante, d'une part et d'autre part tenir compte des limites dues à la mémoire vive. En effet, le langage d'initiation retenu étant le *BASIC interprété*, les lignes à tracer ne doivent pas être trop nombreuses, afin que la succession des vues ne soit pas trop lente. De plus cette méthode est très gourmande en mémoire, celle-ci devant être sacrifiée au profit de la rapidité. Or, plus les lignes composant l'objet seront nombreuses, plus les limites de la mémoire vive seront rapidement atteintes. Ces considérations sont surtout valables pour les micro-ordinateurs bas de gamme, ne disposant que de peu de mémoire utilisateur et dont la vitesse du micro-processeur est assez faible.

Rotation

Le programme 7.1 va se charger de satisfaire à toutes ces exigences. La pyramide ne sera décrite dans les lignes de *DATA* qu'en une seule position de départ, située verticalement. Le logiciel calculera alors les différentes positions occupées successivement par la pyramide, dans l'espace. Car nous allons imprimer à cet objet une **rotation** suivant les trois axes de coordonnées. Il sera donc nécessaire d'appliquer les méthodes précédentes, pour chaque vue, afin de pouvoir éliminer ce qui ne doit pas être tracé. Or, appliquer ces méthodes en temps réel à chaque dessin, aboutirait à des temps de traitement prohibitifs. Par contre une astuce permet de surmonter ce problème: si tous les calculs sont effectués *massivement*, avant de commencer les tracés, ceux-ci ne dureront que le temps mis par le système à tracer des suites de points. C'est-à-dire un temps très court, comparé à celui utilisé par les calculs. C'est ce qu'effectue le programme. Ainsi, **36** vues différentes de la pyramide en rotation sont calculées par avance, selon la méthode des surfaces cachées, appliquée à chacune d'entre elles. Puis les différents tracés commenceront.

HPAGE

Le programme est présenté selon les procédés habituels. Les lignes 110 à 190 reprennent les quelques commandes susceptibles de varier d'un appareil à l'autre. La commande **HPAGE** est très utile: elle permet de visualiser une page, tandis que des tracés seront effectués sur l'autre, donc à l'insu du spectateur. En intervenant judicieusement à ce niveau, des effets intéressants peuvent être obtenus. Parmi les initialisations effectuées notamment en ligne 230, le tableau **NP%** doit être suffisamment dimensionné, dans le but de recevoir toutes les coordonnées nécessaires.




```
10 '**********************************************************
20 '*         ROTATION D'UN OBJET EN 3 DIMENSIONS            *
30 '*        AVEC ELIMINATION DES SURFACES CACHEES           *
40 '**********************************************************
50 '
60 GOTO 340
70 '
80 '             ==================================
90 '============  bloc des commandes specifiques  ===========
100 '            ==================================
110 CLS: RETURN 'Effacement ecran Basse Resolution
120 MX= 320: MY= 250: RETURN 'Surface ecran H.R. (Nbe points)
130 HDISP 1: RETURN 'Visu Basse Resolution
140 HDISP 2: RETURN 'Visu Haute Resolution
150 'HCLS x => Effacement H.R., avec couleur x
160 'HCOLOR x => Couleur du trace (0 a 7)
170 'HPLOT x,y => Trace un point
180 'HPLOT x,y TO x',y' => Trace une droite
190 'HPAGE x,y => H.R. : visualisation page x, ecriture page y
200 '=========================================================
210 '
220 '****************** INITIALISATIONS ******************
230 DIM NP%(912), B1(6,3), B2(4,3), B3(4,2), B4(4,4), B5(4,3)
240 HCOLOR TC 'Init. couleur H.R.
250 CX= INT(MX/2): CY= INT(MY/2) 'Centre ecran
260 PT= 1
270 THETA= THETA*3.14159265/180: PHI= PHI*3.14159265/180
280 TS= SIN(THETA): TE= COS(THETA): PS= SIN(PHI): PC= COS(PHI)
290 A1= -.1: A2= .1: A3= COS(A2): A4= SIN(A2): A5 =SIN(A1): A6= COS(A1)
300 TP= -.1: SP= SIN(TP): PP= COS(TP)
310 RETURN
320 '
```

Programme 7.1 ▶▶▶

Limite supérieure

Pour calculer cette limite supérieure, il convient d'appliquer les calculs suivants: la pyramide montre **6** côtés, chacun déterminé par deux points, possédant deux coordonnées (**x** et **y**). Ce qui nous donne 6×2×2=24 points, par vue. Or, nous désirons 36 vues différentes,ce qui porte le total de points à 864. Mais afin de boucler proprement la succession des 36 vues, en éliminant d'éventuelles dérives dans les calculs, dues à des erreurs d'arrondis, deux vues fictives supplémentaires permettront d'effacer correctement celle d'origine et de boucler le cycle de façon parfaite. Il faut donc rajouter deux fois 24 points au tableau, d'où un total final de 912 points. Les vues 37 et 38 seront les copies conformes des deux premières vues.

▶▶▶

HPAGE sélectionne la page en mémoire n° 0 ou 1, pour y effectuer des tracés. A adapter à votre matériel.

```
330 '****************** DEBUT DU PROGRAMME *******************
340 GOSUB 130: GOSUB 110: GOSUB 120 'Effacement ecran B.R.
350 INPUT "Couleur du fond (0 ... 7) "; CF
360 IF (CF < 0) OR (CF > 7) THEN CF= 0
370 HPAGE 1: HCLS CF: HPAGE 0: HCLS CF
380 INPUT "Couleur du trace (0 ... 7) "; TC
390 IF (TC < 0) OR (TC > 7) THEN TC= 7
400 INPUT "Cadre desire (o,N) "; CD$
410 IF (CD$ = "n") OR (CD$ = "N") THEN 440  ELSE INPUT "Couleur du cadre (0 ... 7) "; CC: HCOLOR
 CC
420 HPAGE 1: HPLOT 0,0 TO MX-1,0: HPLOT MX-1,0 TO MX-1,MY-1: HPLOT MX-1,MY-1 TO 0,MY-1: HPLOT 0,
MY-1 TO 0,0
430 HPAGE 0: HPLOT 0,0 TO MX-1,0: HPLOT MX-1,0 TO MX-1,MY-1: HPLOT MX-1,MY-1 TO 0,MY-1: HPLOT 0,
MY-1 TO 0,0
440 INPUT "Distance de l'observateur RHO (15) "; RHO
450 INPUT "Distance de l'ecran (350) "; D
460 INPUT "Angle THETA en degres (30) "; THETA
470 INPUT "Angle PHI en degres (50) "; PHI
480 '
490 PRINT: PRINT "Calcul des differentes positions ..."
500 GOSUB 230  'Initialisations
510 FOR I= 1 TO 4
520   READ X,Y,Z
530   B2(I,1)= X: B2(I,2)= Y: B2(I,3)= Z
540   XX= -X*TS+Y*TE: YY= -X*TE*PC-Y*TS*PC+Z*PS: ZZ= -X*PS*TE-Y*PS*TS-Z*PC+RHO
550   B3(I,1)= D*(XX/ZZ)+CX: B3(I,2)= -D*(YY/ZZ)+CY
560 NEXT I
570 FOR I= 1 TO 4
580   FOR J= 1 TO 4
590     READ B4(I,J)
600   NEXT J
610 NEXT I
```

L'observateur pourra se déplacer autour de l'objet en répondant aux 4 lignes qui suivent.

C'est à l'ordinateur d'effectuer tout le travail.

Programme 7.1 ▶▶▶

Paramètres

Le début du programme s'effectue en ligne 340, en demandant à l'utilisateur l'entrée de différents paramètres. La couleur du fond est importante, puisque les effacements seront effectués en réalité par des tracés dans cette couleur, recouvrant les anciens. A partir de la ligne 440, jusqu'à la ligne 470, les paramètres concernant le point de vision de l'observateur sont demandés, avec, entre parenthèses, les valeurs conseillées au départ.

Puis des initialisations sont effectuées, transformant en *radians* les angles précisés préalablement en degrés et en calculant une fois pour toutes certaines valeurs (lignes 270 à 300). La phase relative aux calculs des 36 vues commence alors, à partir de la ligne 510, par la prise en compte des coordonnées définissant la pyramide. Puis, de la ligne 570 à la ligne 610, les renseignements relatifs à la description des faces sont lus par l'intermédiaire des lignes de *DATA*.

Ensuite les lignes 620 à 1040 calculent les 36 vues consécutives, en déterminant à chaque fois quels en seront les côtés visibles et invisibles. Le tableau **NP%** contiendra les coordonnées visibles de chaque côté (ligne 910). Puis les lignes 940 à 1020 effectueront les transformations nécessaires aux différentes rotations. Afin de renseigner l'utilisateur sur le temps d'attente nécessaire au calcul des 36 vues, le numéro en cours de chaque vue terminée est affiché ligne 1030 (AG). A la fin des calculs relatifs aux 36 vues, les deux premières sont recopiées dans les vues 37 et 38, pour les raisons précitées (lignes 1050 à 1070).

▶▶▶

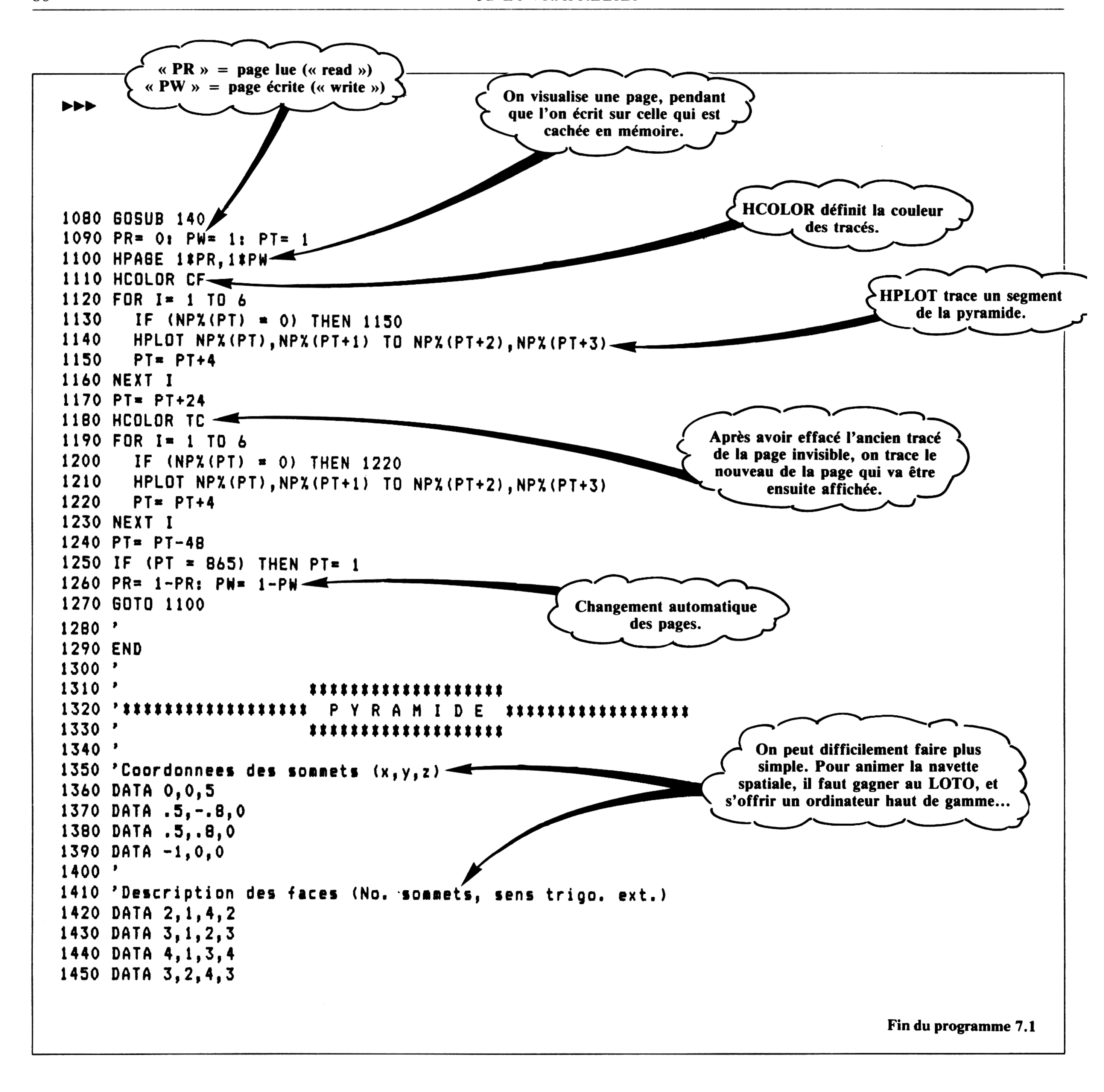
Calculs pour 36 vues.

```
620 FOR AG= 1 TO 36
630    FOR I= 1 TO 6
640      B1(I,3)= 0
650    NEXT I
660    FOR I= 1 TO 4
670      W1= B2(B4(I,2),1)-B2(B4(I,1),1): W2= B2(B4(I,2),2)-B2(B4(I,1),2): W3= B2(B4(I,2),3)-B2(B
4(I,1),3)
680      W4= B2(B4(I,3),1)-B2(B4(I,1),1): W5= B2(B4(I,3),2)-B2(B4(I,1),2): W6= B2(B4(I,3),3)-B2(B
4(I,1),3)
690      B5(I,1)= W2*W6-W5*W3: B5(I,2)= W3*W4-W6*W1: B5(I,3)= W1*W5-W4*W2
700    NEXT I
710    XX= RHO*PS*TE: YY= RHO*PS*TS: ZZ= RHO*PC
720    B5= 1
730    FOR I= 1 TO 4
740      EE= B4(I,1)
750      WX= XX-B2(EE,1): WY= YY-B2(EE,2): WZ= ZZ-B2(EE,3)
760      IF ((B5(I,1)*WX+B5(I,2)*WY+B5(I,3)*WZ) <= 0) THEN 870
770      FF= B4(I,1)
780      FOR J= 2 TO 4
790        EE= B4(I,J)
800        FOR K= 1 TO B5
810          IF (B1(K,1) = EE) AND (B1(K,2) = FF) THEN B1(K,3)= 2: GOTO 850
820        NEXT K
830        B1(B5,1)= FF: B1(B5,2)= EE: B1(B5,3)= 1
840        B5= B5+1
850        FF= EE
860      NEXT J
870    NEXT I
880    FOR I= 1 TO 6
890      IF (B1(I,3) = 0) THEN 920
900      J= B1(I,1): K= B1(I,2)
910      NP%(PT)= B3(J,1): NP%(PT+1)= B3(J,2): NP%(PT+2)= B3(K,1): NP%(PT+3)= B3(K,2)
920      PT= PT+4
930    NEXT I
940    FOR I= 1 TO 4
950      SS= PP*A3*B2(I,1)-(A4*PP+A5*SP)*B2(I,2)+(A5*A4*PP-SP*A6)*B2(I,3)
960      UU= A4*B2(I,1)+A6*A3*B2(I,2)-A5*A3*B2(I,3)
970      VV= SP*A3*B2(I,1)+(A5*PP-A6*A4*SP)*B2(I,2)+(A4*A5*SP+A6*PP)*B2(I,3)
980      B2(I,1)= SS: B2(I,2)= UU: B2(I,3)= VV
990      X= SS: Y= UU: Z= VV
1000      XX= -X*TS+Y*TE: YY= -X*TE*PC-Y*TS*PC+Z*PS: ZZ= -X*PS*TE-Y*PS*TS-Z*PC+RHO
1010      B3(I,1)= D*(XX/ZZ)+CX: B3(I,2)= -D*(YY/ZZ)+CY
1020    NEXT I
1030    PRINT AG
1040 NEXT AG
1050 FOR I= 1 TO 48
1060    NP%(I+864)= NP%(I)
1070 NEXT I
```

Coordonnées visibles de chaque côté.

Recopie des deux premières « vues » dans les vues 37 et 38.

Programme 7.1 ▶▶▶



▶▶▶

```
1080 GOSUB 140
1090 PR= 0: PW= 1: PT= 1
1100 HPAGE 1*PR,1*PW
1110 HCOLOR CF
1120 FOR I= 1 TO 6
1130   IF (NP%(PT) = 0) THEN 1150
1140   HPLOT NP%(PT),NP%(PT+1) TO NP%(PT+2),NP%(PT+3)
1150   PT= PT+4
1160 NEXT I
1170 PT= PT+24
1180 HCOLOR TC
1190 FOR I= 1 TO 6
1200   IF (NP%(PT) = 0) THEN 1220
1210   HPLOT NP%(PT),NP%(PT+1) TO NP%(PT+2),NP%(PT+3)
1220   PT= PT+4
1230 NEXT I
1240 PT= PT-48
1250 IF (PT = 865) THEN PT= 1
1260 PR= 1-PR: PW= 1-PW
1270 GOTO 1100
1280 '
1290 END
1300 '
1310 '                    *******************
1320 '******************  P Y R A M I D E  *****************
1330 '                    *******************
1340 '
1350 'Coordonnees des sommets (x,y,z)
1360 DATA 0,0,5
1370 DATA .5,-.8,0
1380 DATA .5,.8,0
1390 DATA -1,0,0
1400 '
1410 'Description des faces (No. sommets, sens trigo. ext.)
1420 DATA 2,1,4,2
1430 DATA 3,1,2,3
1440 DATA 4,1,3,4
1450 DATA 3,2,4,3
```

Fin du programme 7.1

Page visualisée et page écrite

Enfin, les tracés proprement dits pourront débuter. **PR** et **PW** désignent successivement le numéro de la page *visualisée* et celui de la page *écrite*. Le pointeur des vues (**PT**), est initialisé à **1** (ligne 1090). La ligne 1100 peut alors en déduire alternativement quelle sera la page vue et quelle sera celle écrite. Les lignes 1180 à 1230 tracent les côtés de la pyramide, en couleur visible. Des lignes 1110 à 1160, les côtés précédemment tracés seront effacés, par retraçage de ces mêmes lignes, mais cette fois à l'aide de la couleur du fond. Les lignes 1140 et 1210 effectuent les tracés proprement dits, à partir des coordonnées contenues dans le tableau **NP%**, après mise à jour du pointeur, de façon adéquate (lignes 1150, 1170, 1220 et 1240).

Changements de page

A la ligne 1260, le changement de page est effectué : celle qui était vue devient celle qui sera écrite et vice versa. C'est ce qui permet l'effet de mouvement, car le changement de page n'est pas perçu par le spectateur, tandis que les différents tracés et effacements s'effectuent sur la page invisible. Ainsi, à chaque mouvement, correspondent deux tracés sur la page écrite : l'effacement du pré-

Rotation en 36 vues

cédent et la génération du suivant, alors que le premier résultat est visualisé en ce moment même. Puis la page pivote et le cycle recommence sur l'autre page. Et ainsi de suite, se succèdent alternativement les 36 vues suggérant la rotation de la pyramide. Les lignes 1360 à 1450 contiennent les *DATA*, définies comme étudié lors des programmes précédents.

3. LES FRACTALES EN TROIS DIMENSIONS

Troisième dimension

En poursuivant les recherches relatives à la *troisième dimension*, une nouvelle voie peut être explorée, assez différente de celles déjà étudiées jusqu'à maintenant. Elle consiste à adapter à cette dimension les dessins relatifs aux courbes *fractales*, généralement construites au sein de la *deuxième dimension.*

Courbes fractales

Le programme 7.2 utilise une technique décrite dans la revue « Scientific American » (septembre 1984, U.S.A). Les dessins produits formeront des paysages composés de montagnes et de mers. Cela aléatoirement, en suivant les lois de la nature, mais surtout en conférant à l'ensemble du paysage une certaine cohérence naturelle. Les courbes fractales permettent d'obtenir des simulations réelles de phénomènes naturels, à partir de formes géométriques simples, dont le motif peut se répéter à l'infini en utilisant des échelles de plus en plus petites.



mais cohérente !

Chaque fraction du motif est elle-même remplacée par l'intégralité de celui-ci et ainsi de suite, jusqu'à obtenir l'effet désiré. C'est ainsi que naquirent les célèbres *dragons* et *flocons* de **Von Koch**.

```
10 '**********************************************************
20 '********  COURBES FRACTALES EN TROIS DIMENSIONS  *********
30 '**********************************************************
40 '
50 GOTO 220
60 '
70 '          ================================
80 '===========  bloc des commandes specifiques  ============
90 '          ================================
100 '
110 CLS: RETURN                'Effacement ecran Basse Resolution
120 XM= 319: YM= 249: RETURN      'Coordonnees max. ecran H.R.
130 ZX= .04: ZY= .04: SZ= .04: RETURN      'Facteur d'echelle
140 CG= 17: CH= 35: RETURN                       'Pour centrage
150 HDISP 1: RETURN                   'Visu Basse Resolution
160 HDISP 2: RETURN                   'Visu Haute Resolution
170 HCOLOR BL: RETURN           'couleur bleue, pour le trace
180 HCOLOR VE: RETURN           'couleur verte, pour le trace
190 'HPLOT x,y TO x',y'                  => Trace une droite
200 '=========================================================
210 '
```

Bloc à adapter selon le type du micro-ordinateur.

Programme 7.2 ▶▶▶

Ces limites sont introduites par la taille de la mémoire.

Il vaut mieux prévenir, c'est assez long...

Ainsi on peut s'assurer de l'avancement des travaux...

Ce qui suit introduit des variations aléatoires au niveau des sommets des triangles.

```
▶▶▶

220 DEFINT A-N
230 DIM NI(128,64)                  'Pour 6 niveaux max => (64,32)
240 GOSUB 150: GOSUB 110                   'Effacement ecran B.R.
250 INPUT "Nombre de niveaux (7 maximum) : "; NN
260 IF (NN < 0) THEN NN= 0                           'Protections
270 IF (NN > 7) THEN NN= 7
280 PRINT: PRINT "Calculs en cours ..."
290 GOSUB 120: BL= 4: VE= 7: GOSUB 140           'Initialisations
300 CE= 2
310 FOR N= 1 TO NN
320   CE= CE+2^(N-1)
330 NEXT N
340 CL= CE-1: DL= CL/2
350 PI= 3.14159: RHO= PI*30/180: SO= RHO*1.2
360 PRINT
370 FOR N= 1 TO NN
380   L= 10000/1.8^N
390   PRINT "Niveau traite : " N
400   HC= CL/2^N: TJ= HC*2
410   GOSUB 480    'entre en table hauteurs dans le sens des X
420   GOSUB 630    'entre en table hauteurs dans le sens des Y
430   GOSUB 780    'entre en table hauteurs dans le sens diag.
440 NEXT N
450 GOTO 1390                              'Commence le trace
460 '
470 '************ hauteurs, dans le sens des X  ************
480 FOR RB= 0 TO (CL-1) STEP TJ
490   FOR SB= (HC+RB) TO CL STEP TJ
500     MR= SB-HC: MS= RB
510     GOSUB 940
520     ST= NI: MR= SB+HC
530     GOSUB 940
540     SU= NI
550     NI= (ST+SU)/2+RND(L/2)-L/4
560     MR= SB: MS= RB
570     GOSUB 1020
580   NEXT SB
590 NEXT RB
600 RETURN
610 '
620 '************ hauteurs, dans le sens des Y *************
630 FOR SB= CL TO 1 STEP -TJ
640   FOR RB= HC TO SB STEP TJ
650     MR= SB: MS= RB+HC
660     GOSUB 940
670     ST= NI: MS= RB-HC
680     GOSUB 940
690     SU= NI
700     NI= (ST+SU)/2+RND(L/2)-L/4
710     MR= SB: MS= RB
720     GOSUB 1020
730   NEXT RB
740 NEXT SB
750 RETURN
```

Programme 7.2 ▶▶▶

▶▶▶

```
760 '
770 '************  hauteurs, dans le sens diag.  *************
780 FOR SB= 0 TO (CL-1) STEP TJ
790   FOR RB= HC TO (CL-SB) STEP TJ
800     MR= SB+RB-HC: MS= RB-HC
810     GOSUB 940
820     ST= NI
830     MR= SB+RB+HC: MS= RB+HC
840     GOSUB 940
850     SU= NI
860     MR= SB+RB: MS= RB
870     NI= (ST+SU)/2+RND(L/2)-L/4
880     GOSUB 1020
890   NEXT RB
900 NEXT SB
910 RETURN
920 '
930 '**********  lecture des donnees de la table  ************
940 IF (MS > DL) THEN 970
950 CR= MR: CS= MS
960 GOTO 980
970 CR= CL-MR: CS= CL+1-MS
980 NI= NI(CR,CS)
990 RETURN
1000 '
1010 '**********  chargement des donnees en table  ***********
1020 IF (MS > DL) THEN 1050
1030 CR= MR: CS= MS
1040 GOTO 1060
1050 CR= CL-MR: CS= CL+1-MS
1060 NI(CR,CS)= NI
1070 RETURN
1080 '
1090 '**********  prise en compte du niveau "mer"  ***********
1100 IF (TT <> -999) THEN 1140
1110 IF (A3 < 0) THEN GOSUB 170: Z2= A3: A3= 0: GOTO 1350
1120 GOSUB 180
1130 GOTO 1340
1140 IF (Z2 > 0) AND (A3 > 0) THEN 1340
1150 IF (Z2 < 0) AND (A3 < 0) THEN Z2= A3: A3= 0: GOTO 1350
1160 W3= A3/(A3-Z2)
1170 D1= (E1-A1)*W3+A1: D2= (E2-A2)*W3+A2: D6= 0
1180 XT= A1: D4= A2: D5= A3
1190 IF (A3 > 0) THEN 1300
1200 '
1210 '******************  trace la "mer"  ********************
1220 A1= D1: A2= D2: A3= D6
1230 GOSUB 1900
1240 GOSUB 170
1250 A1= XT: A2= D4: A3= 0
1260 Z2= D5
1270 GOTO 1350
1280 '
```

C'est beaucoup plus joli avec la mer, mais certains préfèrent la montagne...

Si l'utilisateur l'a demandé

Programme 7.2 ▶▶▶

La figure 7.4 montre le résultat obtenu, sous forme d'une île montagneuse dont on perçoit même les vagues de la mer qui l'entoure. Afin de comprendre le principe utilisé, reportez vous à la figure 7.3, montrant une autre génération de ce type de paysage mais à un degré inférieur. Le tracé fait apparaître une trame ténue de dessins géométriques, plus ou moins torturés.

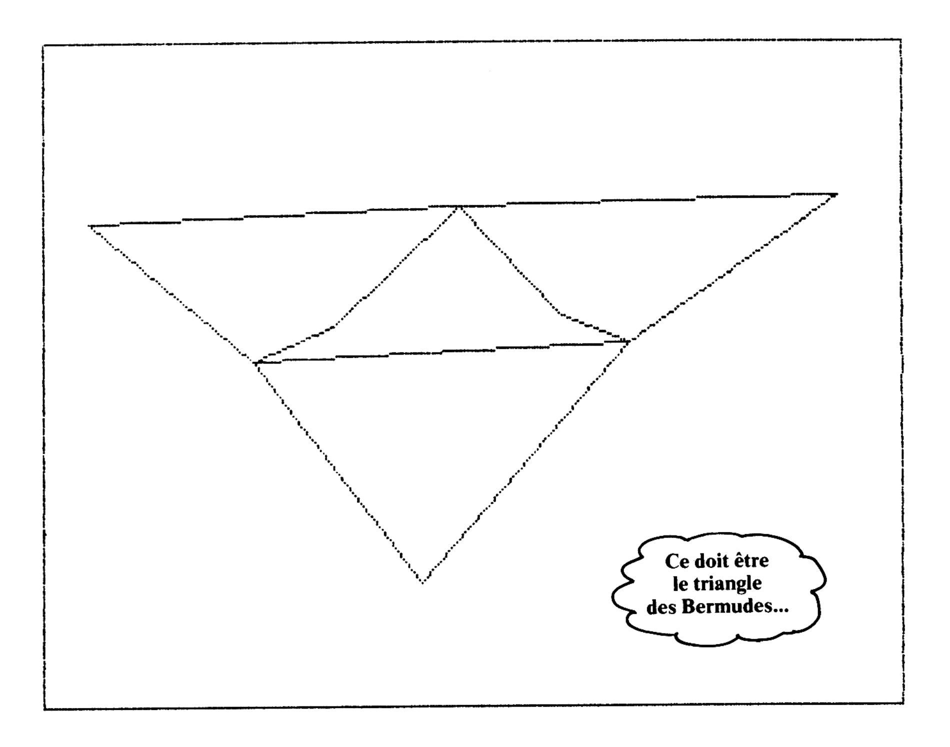


Figure 7.1 — Figure de base générant des paysages en courbes fractales à trois dimensions. Le degré « 1 » trace quatre triangles, dont la position occupée par les sommets est aléatoire.

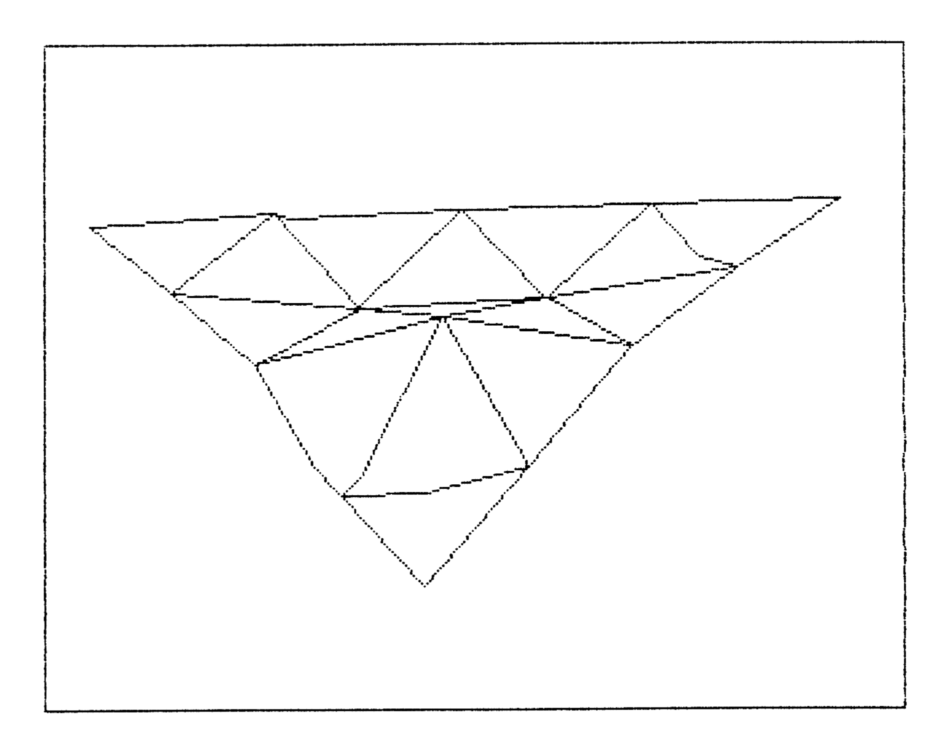


Figure 7.2 — Augmentation du degré « 2 », d'une courbe fractale en trois dimensions. 16 triangles ont été générés.

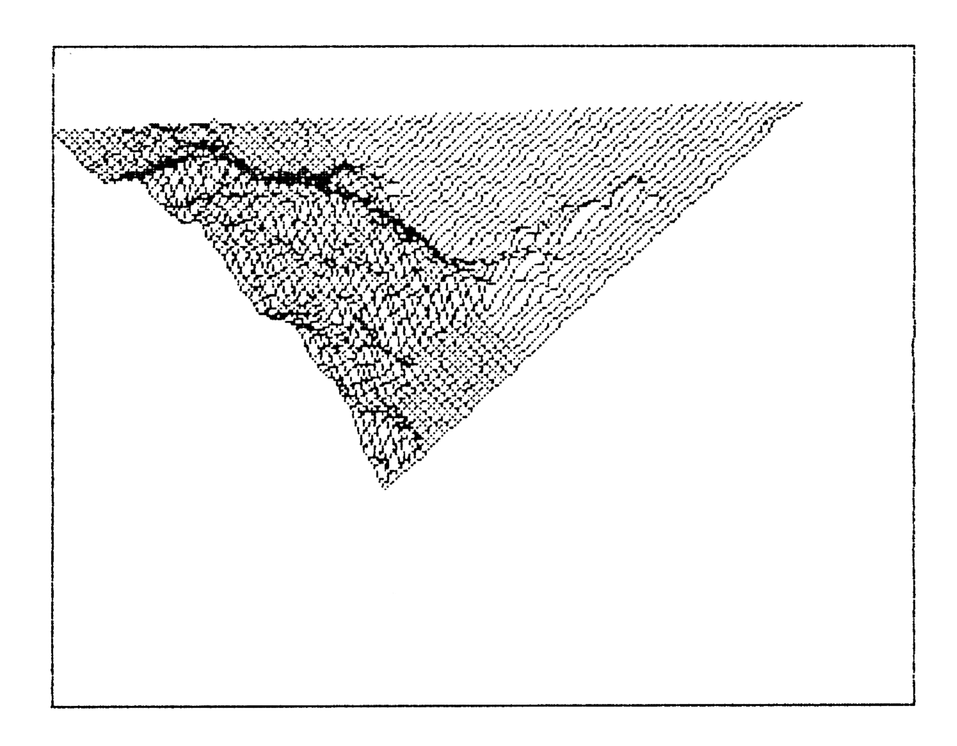


Figure 7.3 — Poursuite de la recherche en finesse des courbes fractales en trois dimensions. Le degré sélectionné était d'ordre « 6 », permettant le tracé de 4496 triangles aléatoires.

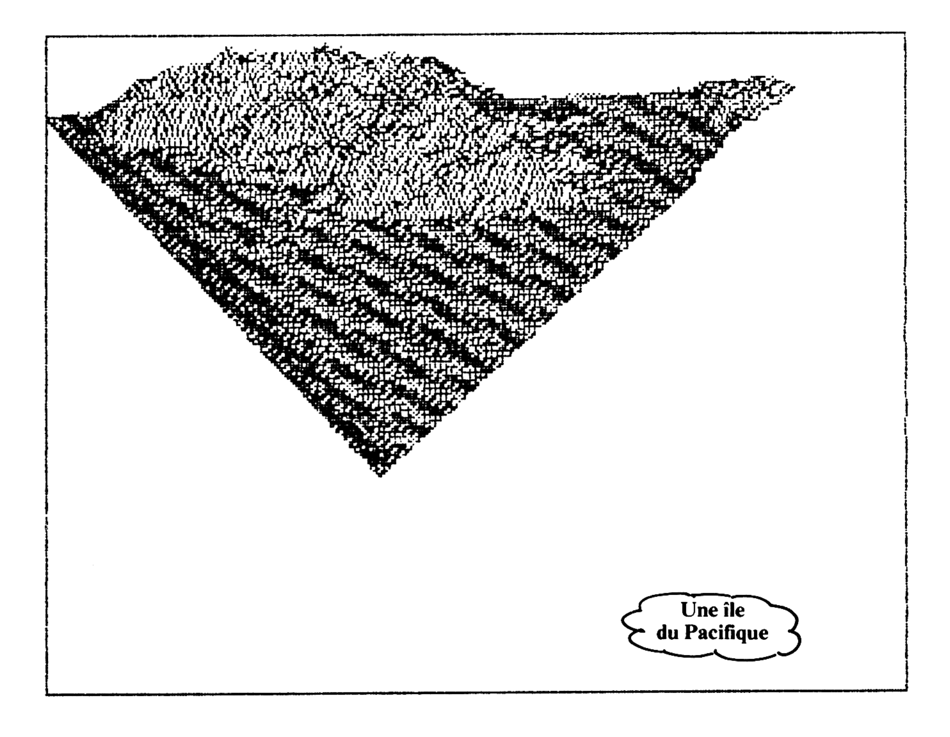


Figure 7.4 — Pratiquement le maximum toléré par la définition de l'écran. Le degré « 7 » a créé, d'une façon aléatoire, 16384 triangles.

En poursuivant notre progression à l'envers, la figure 7.2 possède un degré de réalisation suffisamment faible pour montrer le détail du dessin : à savoir un ensemble de triangles légèrement déformés. Enfin la figure 7.1 met en évidence la forme géométrique de base employée. Un triangle constitue la figure d'origine, contenant un autre triangle dont les sommets sont situés au milieu des côtés du

```
▶▶▶
1290 '**************** trace la "montagne" ****************
1300 A1= D1: A2= D2: A3= D6
1310 GOSUB 1900
1320 GOSUB 180
1330 A1= XT: A2= D4: A3= D5
1340 Z2= A3
1350 E1= A1: E2= A2
1360 RETURN
1370 '
1380 '********************** dessin **********************
1390 GOSUB 160                'selection du mode Haute Resolution
1400 GOSUB 130                                           'echelle
1410 FOR MR= 0 TO CL
1420   TT= -999
1430   FOR MS= 0 TO MR
1440     GOSUB 940
1450     A3= NI: A2= MS/CL*10000: A1= MR/CL*10000-A2/2
1460     GOSUB 1890
1470   NEXT MS
1480 NEXT MR
1490 FOR MS= 0 TO CL
1500   TT= -999
1510   FOR MR= MS TO CL
1520     GOSUB 940
1530     A3= NI: A2= MS/CL*10000: A1= MR/CL*10000-A2/2
1540     GOSUB 1890
1550   NEXT MR
1560 NEXT MS
1570 FOR EX= 0 TO CL
1580   TT= -999
1590   FOR EY= 0 TO (CL-EX)
1600     MR= EX+EY: MS= EY
1610     GOSUB 940
1620     A3= NI: A2= MS/CL*10000: A1= MR/CL*10000-A2/2
1630     GOSUB 1890
1640   NEXT EY
1650 NEXT EX
1660 END '                                       tracage termine
1670 '
1680 '********************** rotation *********************
1690 IF (A1 <> 0) THEN 1730
1700 IF (A2 <= 0) THEN TB= -PI/2: GOTO 1750
1710 TB= PI/2
1720 GOTO 1750
1730 TB= ATN(A2/A1)
1740 IF (A1 < 0) THEN TB= TB+PI
1750 R1= TB+RHO: DJ= SQR(A1*A1+A2*A2)
1760 A1= DJ*COS(R1): A2= DJ*SIN(R1)
1770 RETURN
1780 '
1790 '************* trace, jusqu'au-dessous **************
1800 DJ= SQR(A3*A3+A1*A1)
1810 IF (A1 = 0) THEN TB= PI/2: GOTO 1840
1820 TB= ATN(A3/A1)
1830 IF (A1 < 0) THEN TB= TB+PI
```

Le tracé commence enfin ! Mais tous les calculs sont effectués.

Pour tenir compte de l'éventuelle présence de « la mer ».

Programme 7.2 ▶▶▶

▶▶▶

```
1840 R1= TB-SO
1850 A1= DJ*COS(R1)+A1: A3= DJ*SIN(R1)
1860 RETURN
1870 '
1880 '**********  trace ou se deplace a (WX,WY)  ***********
1890 GOSUB 1100
1900 A1= A1*ZX: A2= A2*ZY: A3= A3*SZ
1910 GOSUB 1690                                    'rotation
1920 GOSUB 1800                          'trace, jusqu'au-dessus
1930 IF (TT = -999) THEN WR$= "M"  ELSE WR$= "D"
1940 WX= INT(A2)+CX: WY= INT(A3)
1950 GOSUB 1990
1960 RETURN
1970 '
1980 '***************  trace d'une ligne  ****************
1990 WX= WX*.625: WY= 33.14-.663*WY
2000 IF (WR$ = "M") THEN MM= WX: PP= WY: TT= X
2010 IF (PP > YM) OR (PP < 0) OR (WY > YM) OR (WY < 0) THEN RETURN
2020 HPLOT (MM+CG),(PP+CH) TO (WX+CG),(WY+CH)
2030 MM= WX: PP= WY
2040 RETURN
2050 '
2060 '=====================================================
2070 '=====================================================
```

Trace un petit segment de droite. A adapter selon le langage utilisé.

fin du programme 7.2

premier. Par contre, ces trois sommets vont subir des variations aléatoires au niveau de leurs coordonnées, dans le sens vertical. C'est cet effet qu'affiche la figure, provoquant des déformations spécifiques à chaque tracé. Par convention, ce niveau reçoit le degré « 1 ». Afin de passer au degré « 2 », il suffit de tracer un nouveau triangle, à l'intérieur de chacun des quatre triangles formés par la figure de base. Tout en respectant pour tout sommet le principe de la variation aléatoire des coordonnées et ainsi de suite pour les degrés suivants.

Chaque fois que ce procédé est répété, le nombre de triangles quadruple. Pour le degré « 1 » de la figure 7.1, quatre triangles sont générés. Par contre la figure 7.2 est formée de 16 triangles, relatifs au degré « 2 ». Donc le degré « 3 » dessinerait 64 triangles etc. La figure 7.3 concerne le degré « 6 », donc 4 496 triangles ont été tracés. Enfin la figure 7.4, de degré « 7 », génère 16 384 triangles. C'est à peu près le degré maximum que l'on peut obtenir sur un micro-ordinateur classique. A ce moment les triangles sont tellement petits, qu'ils atteignent la taille d'un *pixel*. D'autre part, la taille mémoire utilisée par les différents calculs est assez importante et le temps total nécessaire à l'obtention de 16 384 triangles est loin d'être négligeable.

Une particularité intéressante a été ajoutée à cette technique: c'est celle permettant la génération automatique d'un niveau « mer ». En attribuant une couleur **verte** aux massifs montagneux et **bleue** à la mer, les effets obtenus sont saisissants de réalisme avec le niveau « 7 ». Ce niveau « mer » sert de référence pour ce qui doit être tracé. Tout ce qui se trouve au-dessus est dessiné à l'aide de la technique décrite plus haut, en vert pour le niveau « montagne », par exemple. Mais dès que le niveau de la mer est atteint, c'est celle-ci qui est tracée, en bleu, jusqu'à ce que les effets du hasard redonnent la prédominance au niveau de la montagne.

L'utilisation du programme est très simple: une seule question est posée, relative au niveau désiré (7 au maximum). Sa présentation suit le schéma classiquement employé depuis le début. Les commandes spécifiques sont regroupées en tête, des lignes 110 à 190. En ligne 230, un tableau est défini (**NI**), dont les caractéristiques varient en fonction du degré de sophistication du matériel utilisé. Pour un maximum de **6** niveaux, une définition de (64,32) suffit. Mais un septième niveau nécessite une dimension de (128,64). Et ainsi de suite pour d'autres niveaux, marquant la limite possible du degré de finesse des tracés. Il est possible d'annuler le niveau « mer », en remplaçant le contenu de la ligne 1 100 par la commande « RETURN ».

Afin d'obtenir des effets intéressants, il est nécessaire d'utiliser au moins le niveau **5**. Naturellement, un choix doit être effectué entre la durée d'exécution et la perfection des résultats obtenus. Certains spécialistes utilisant du matériel haut de gamme, ont réussi à synthétiser des paysages semblant issus de reproductions photographiques. Ce domaine de la recherche graphique, assez récemment découvert, reste ouvert à diverses investigations. Il peut être tentant de débuter les calculs par toute autre figure géométrique.

CHAPITRE 8

LA C.A.O.

Au fil des chapitres précédents, un ensemble de plus en plus complet de fonctions logicielles a vu le jour, progressivement. La complexité a été étendue petit à petit bien entendu, mais surtout et c'est l'essentiel, la puissance fonctionnelle s'est étoffée régulièrement. Pourtant, même si le problème de l'effacement automatique des différentes lignes et surfaces cachées a trouvé des solutions, c'était au prix de quelques limitations de conception. Or il est possible d'aller encore plus loin au niveau des représentations graphiques. Mais pour cela, étant donné le manque de puissance des machines actuellement disponibles sur le marché, il sera nécessaire d'abandonner pour l'instant l'effacement des lignes cachées, du moins automatiquement. La perte, sur le plan de l'automatisation, sera toutefois largement compensée par l'augmentation extraordinaire de la puissance de création.

C'est ainsi que les différents domaines de la **C.A.O.**, c'est-à-dire la ***Conception Assistée par Ordinateur***, vont pouvoir être abordés sans limitation. Une fois le résultat obtenu suffisamment parfait, il sera toujours possible d'effectuer les retouches nécessaires, afin de peaufiner le dessin. Enfin quelques **animations** viendront optionnellement agrémenter ces créations.

1. LE PROGRAMME

S'il est aisé pour chacun d'entre nous de tracer les plans d'objets de toutes sortes, vus de face, il en va tout autrement lorsque ceux-ci doivent être montrés sous un certain angle, comme c'est le cas le plus général dans la réalité. Tout le monde n'est pas un dessinateur talentueux et le plus souvent des distorsions disgracieuses viennent détruire l'excellente idée de départ. D'autant plus qu'à notre époque, la qualité de présentation devient de plus en plus primordiale. Il peut donc être réellement intéressant de se cantonner justement au tracé de ces dessins simplifiés et de laisser le soin à un micro-ordinateur de résoudre les problèmes relatifs à leur représentation *artistique*, **réelle**.

Traçage de face

C'est justement le but du programme 8.1. L'utilisateur devra, dans un premier temps, tracer les différentes faces de l'objet qu'il étudie, sur du **papier millimétré**, telles qu'elles se présentent dans le cas d'une vision *de face*. Puis les différentes coordonnées seront transcrites en lignes de *DATA*, pour permettre au programme de les « digérer ». C'est lui qui se chargera ensuite d'assembler les faces entres elles et de représenter l'objet ainsi créé, sous tous les angles possibles.

Représentation tridimensionnelle

Trois dimensions et vision spatiale

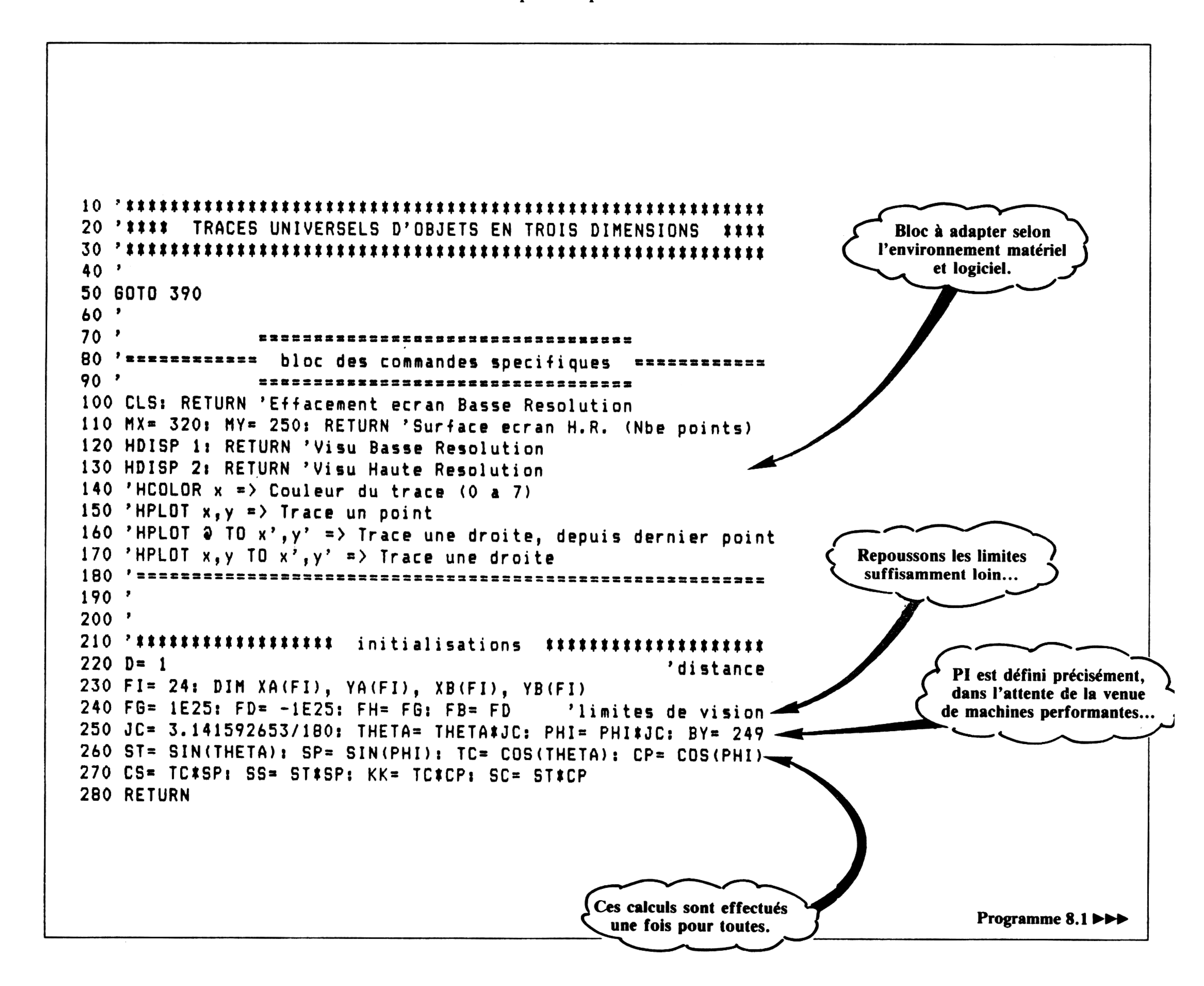
L'utilisateur pourra largement exercer toute sa fantaisie à ce niveau. En respectant naturellement les lois du dessin en *trois dimensions* et de la *vision spatiale.* Afin de simplifier l'entrée des données d'une part et de permettre l'obtention de toutes sortes de tracés, même très « torturés », une option a été préprogrammée. C'est celle qui permet de tracer simplement des **cercles** ou des **arcs de cercle**, dans n'importe quel plan et de n'importe quelle façon.

Cercles, arcs de cercles et cylindres

Le programme résulte de ce qui a été étudié précédemment. Rappelons, afin de préserver un maximum de compatibilité entre les machines disponibles sur le marché, que la *présentation* du logiciel sur l'écran a été réduite au strict minimum, comme pour les autres programmes de ce livre. Donc il n'est pas prévu de menus déroulants, de sous-écrans etc., pour l'entrée des données. Le lecteur connaissant parfaitement son matériel — et celui-ci constituant d'ailleurs le meilleur choix possible — pourra à sa guise insérer des cadres, sous-écrans, surimpressions, inversions vidéo, clignotements ou autres, dans le but de donner à son logiciel la « touche professionnelle », sans laquelle tout programme moderne est voué au rejet immédiat.

Champ de vision

Au début, se trouve le traditionnel bloc des *commandes spécifiques* à chaque micro-ordinateur (lignes 100 à 170). A la ligne 220, la distance éran a été fixée arbitrairement à **1**, car le logiciel se charge automatiquement de cadrer le tracé au mieux de la surface fixée sur l'écran. Dans ce cas, cette distance n'est donc plus significative. Il en est de même pour les limites du *champ de vision*, qui seront en fait imposées par l'utilisateur à l'aide de celles de la « clôture ».



```
10 '************************************************************
20 '****  TRACES UNIVERSELS D'OBJETS EN TROIS DIMENSIONS  ****
30 '************************************************************
40 '
50 GOTO 390
60 '
70 '              ===================================
80 '============  bloc des commandes specifiques  ============
90 '              ===================================
100 CLS: RETURN 'Effacement ecran Basse Resolution
110 MX= 320: MY= 250: RETURN 'Surface ecran H.R. (Nbe points)
120 HDISP 1: RETURN 'Visu Basse Resolution
130 HDISP 2: RETURN 'Visu Haute Resolution
140 'HCOLOR x => Couleur du trace (0 a 7)
150 'HPLOT x,y => Trace un point
160 'HPLOT @ TO x',y' => Trace une droite, depuis dernier point
170 'HPLOT x,y TO x',y' => Trace une droite
180 '===========================================================
190 '
200 '
210 '*****************  initialisations  *******************
220 D= 1                                              'distance
230 FI= 24: DIM XA(FI), YA(FI), XB(FI), YB(FI)
240 FG= 1E25: FD= -1E25: FH= FG: FB= FD      'limites de vision
250 JC= 3.141592653/180: THETA= THETA*JC: PHI= PHI*JC: BY= 249
260 ST= SIN(THETA): SP= SIN(PHI): TC= COS(THETA): CP= COS(PHI)
270 CS= TC*SP: SS= ST*SP: KK= TC*CP: SC= ST*CP
280 RETURN
```


Programme 8.1 ▶▶▶

▶▶▶

```
290 '
300 '*********** recherche des coordonnees ecran ***********
310 OX= (-XX*ST)+(YY*TC): OY= (-XX*CS)-(YY*SS)+(ZZ*CP): OZ= (-XX*KK)-(YY*SC)-(ZZ*SP)+RHO
320 XD= D*OX/OZ: YD= D*OY/OZ
330 RETURN
340 '
350 XF= (XD-FG)*UU+CG: YF= BY-((YD-FH)*UU+CH)
360 RETURN
370 '
380 '***************** DEBUT DU PROGRAMME *****************
390 GOSUB 110: GOSUB 120: GOSUB 100      'Effacement ecran B.R.
400 INPUT "Limite GAUCHE fenetre   (6) "; CG
410 INPUT "Limite DROITE fenetre (313) "; CD
420 INPUT "Limite HAUTE fenetre    (6) "; CH
430 INPUT "Limite BASSE fenetre  (243) "; CB
440 IF (CG < 4) THEN CG= 4                          'Securite
450 IF (CD > MX-5) THEN CD= MX-5
460 IF (CH < 4) THEN CH= 4
470 IF (CB > MY-5) THEN CB= MY-5
480 INPUT "Cadre desire (o/N) "; NT$
490 IF (NT$="N") OR (NT$="n") THEN 530
500 VG= CG-3: VD= CD+3: VH= CH-3: VB= CB+3
510 INPUT "Couleur du cadre  (0 ... 7) "; CC: HCOLOR CC
520 HPLOT VG,VH TO VD,VH: HPLOT VD,VH TO VD,VB: HPLOT VG,VB TO VD,VB: HPLOT VG,VH TO VG,VB
530 RE= CH: CH= MY-1-CB: CB= MY-1-RE          'pour type ecran
540 INPUT "Couleur du trace  (0 ... 7) "; CT
550 HCOLOR CT: PRINT
560 INPUT "Distance de l'observateur RHO (400) "; RHO
570 INPUT "Angle THETA en degres          (60) "; THETA
580 INPUT "Angle PHI en degres            (20) "; PHI
590 PRINT: PRINT "Calculs en cours ..."
600 '
610 GOSUB 220                                 'Initialisations
620 '
```

C'est parti !

Les renseignements qui suivent vont permettre d'effectuer réellement toutes les simulations possibles.

Une fois les volontés de l'utilisateur prises en compte, il ne reste plus qu'à attendre...

Programme 8.1 ▶▶▶

Fenêtre

Par la suite, il sera donc possible d'appeler « fenêtre » la partie de l'écran fixant les limites du dessin. Puis les calculs répétitifs seront effectués une fois pour toutes, des lignes 260 à 270. Précisons qu'à la ligne 250, la définition de « **pi** » comporte beaucoup trop de chiffres significatifs, pour des calculs en simple précision. Le « terrain » est seulement préparé, en vue de la venue sur le marché de machines de plus en plus puissantes.

Distance de l'observateur site et azimut .

Le programme débute réellement à partir de la ligne 390. Des lignes 400 à 550, l'utilisateur devra entrer les limites qu'il désire attribuer à la *clôture*, c'est-à-dire à une sorte de *fenêtre*, considérée comme un écran dans l'écran, que les tracés ne doivent franchir sous aucun prétexte. Afin de matérialiser ce sous-écran, un cadre peut être demandé. Les couleurs, choisies pour le cadre éventuel et le tracé, sont définies par l'utilisateur. Puis les caractéristiques propres à chaque dessin seront demandées, avec les différentes options propres aux choix de la **distance de l'observateur** et des angles de **site** et d'**azimut** (lignes 560 à 580).

Cadrage automatique

Une première lecture des données contenues dans les lignes de *DATA*, a pour objectif de déterminer les coordonnées maximales de l'objet à tracer sur l'écran. De la sorte, le cadrage sera calculé pour que le dessin occupe la majorité de la fenêtre active. Les lignes 640 à 810 résolvent ce problème, en déterminant « UU », coefficient permettant d'intervenir sur l'échelle de la représentation. Une seconde lecture sera donc nécessaire (ligne 850), pour effectuer le tracé lui-même. Le reste du programme est classique, avec détermination des différents plans sur lesquels se situent les droites et les courbes, en cours de tracé.

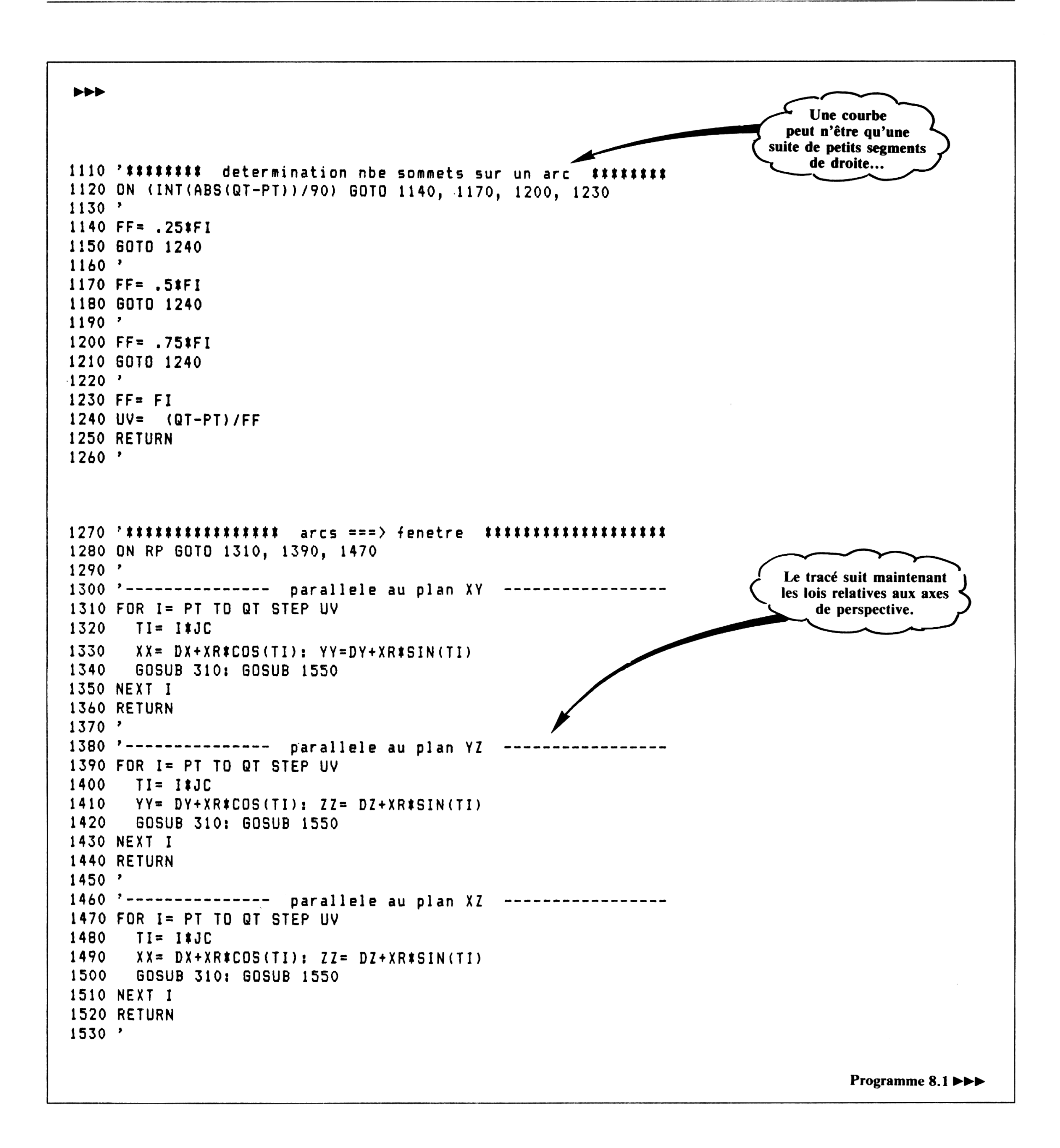
▶▶▶

```
630 '************ premiere lecture ===> fenetre ************
640 READ XX,YY,ZZ,NP
650 ON NP GOTO 670, 700, 740, 790
660 '
670 GOSUB 310: GOSUB 1550
680 GOTO 640
690 '
700 READ XR,RP,PT,QT
710 GOSUB 1120: GOSUB 1280
720 GOTO 640
730 '
740 READ XR,RP
750 PT= 0: QT= 360
760 GOSUB 1120: GOSUB 1280
770 GOTO 640
780 '
790 RU= (CD-CG)/(FD-FG): RV= (CB-CH)/(FB-FH)
800 UU= RU
810 IF (RV < RU) THEN UU= RV
820 '
830 '************** seconde lecture ===> trace **************
840 RESTORE: GOSUB 130          'repositionnement et mode H.R.
850 READ XX,YY,ZZ,NP
860 GOSUB 310
870 ON NP GOTO 930, 970, 1010, 2320
880 '
890 GOSUB 350
900 HPLOT XF,YF
910 GOTO 850
920 '
930 GOSUB 350
940 HPLOT @ TO XF,YF
950 GOTO 850
960 '
970 READ XR,RP,PT,QT
980 GOSUB 1120: GOSUB 1620
990 GOTO 850
1000 '
1010 READ XR,RP
1020 XY$= "T": PP= PP+1
1030 IF (PP = 1) THEN 1070
1040 FOR I= 1 TO FF
1050   XB(I)= XA(I): YB(I)= YA(I)
1060 NEXT I
1070 PT= 0: QT= 360
1080 GOSUB 1120: GOSUB 1620: XY$= ""
1090 GOTO 850
1100 '
```

HPLOT trace ici un point.

Mais à cet endroit, il trace une ligne, à partir du dernier point tracé.

Programme 8.1 ▶▶▶




```
▶▶▶

1110 '******** determination nbe sommets sur un arc ********
1120 ON (INT(ABS(QT-PT))/90) GOTO 1140, 1170, 1200, 1230
1130 '
1140 FF= .25*FI
1150 GOTO 1240
1160 '
1170 FF= .5*FI
1180 GOTO 1240
1190 '
1200 FF= .75*FI
1210 GOTO 1240
1220 '
1230 FF= FI
1240 UV=  (QT-PT)/FF
1250 RETURN
1260 '

1270 '**************** arcs ===> fenetre *******************
1280 ON RP GOTO 1310, 1390, 1470
1290 '
1300 '--------------- parallele au plan XY ----------------
1310 FOR I= PT TO QT STEP UV
1320   TI= I*JC
1330   XX= DX+XR*COS(TI): YY=DY+XR*SIN(TI)
1340   GOSUB 310: GOSUB 1550
1350 NEXT I
1360 RETURN
1370 '
1380 '--------------- parallele au plan YZ ----------------
1390 FOR I= PT TO QT STEP UV
1400   TI= I*JC
1410   YY= DY+XR*COS(TI): ZZ= DZ+XR*SIN(TI)
1420   GOSUB 310: GOSUB 1550
1430 NEXT I
1440 RETURN
1450 '
1460 '--------------- parallele au plan XZ ----------------
1470 FOR I= PT TO QT STEP UV
1480   TI= I*JC
1490   XX= DX+XR*COS(TI): ZZ= DZ+XR*SIN(TI)
1500   GOSUB 310: GOSUB 1550
1510 NEXT I
1520 RETURN
1530 '
```

Programme 8.1 ▶▶▶

Cercle ou polygone, question de définition

Tangentes automatiques

Quelques options ont été prises, dans le but d'optimiser les calculs et les dessins. Par exemple un cercle sera en réalité un polygone de **24** côtés, ce qui est largement suffisant, étant donné la définition des écrans actuels. Mais chacun pourra modifier cette valeur, selon la puissance du matériel qu'il possède, en modifiant « **FI** » à la ligne 230. Les lignes 1120 à 1250 déterminent, le cas échéant, le nombre de côtés du polygone à tracer, dans le cas de la génération d'un arc de cercle. Enfin, toujours dans le domaine des tracés circulaires, si deux cercles se suivent, leurs tangentes seront générées **automatiquement**. Cette option est précieuse, pour représenter simplement des volumes cylindriques (« **XY$="T"** », à la ligne 1020).

▶▶▶

```
1540 '*********** recherche fenetre de projection ***********
1550 IF (XD < FG) THEN FG= XD
1560 IF (XD > FD) THEN FD= XD
1570 IF (YD < FH) THEN FH= YD
1580 IF (YD > FB) THEN FB= YD
1590 RETURN
1600 '
```

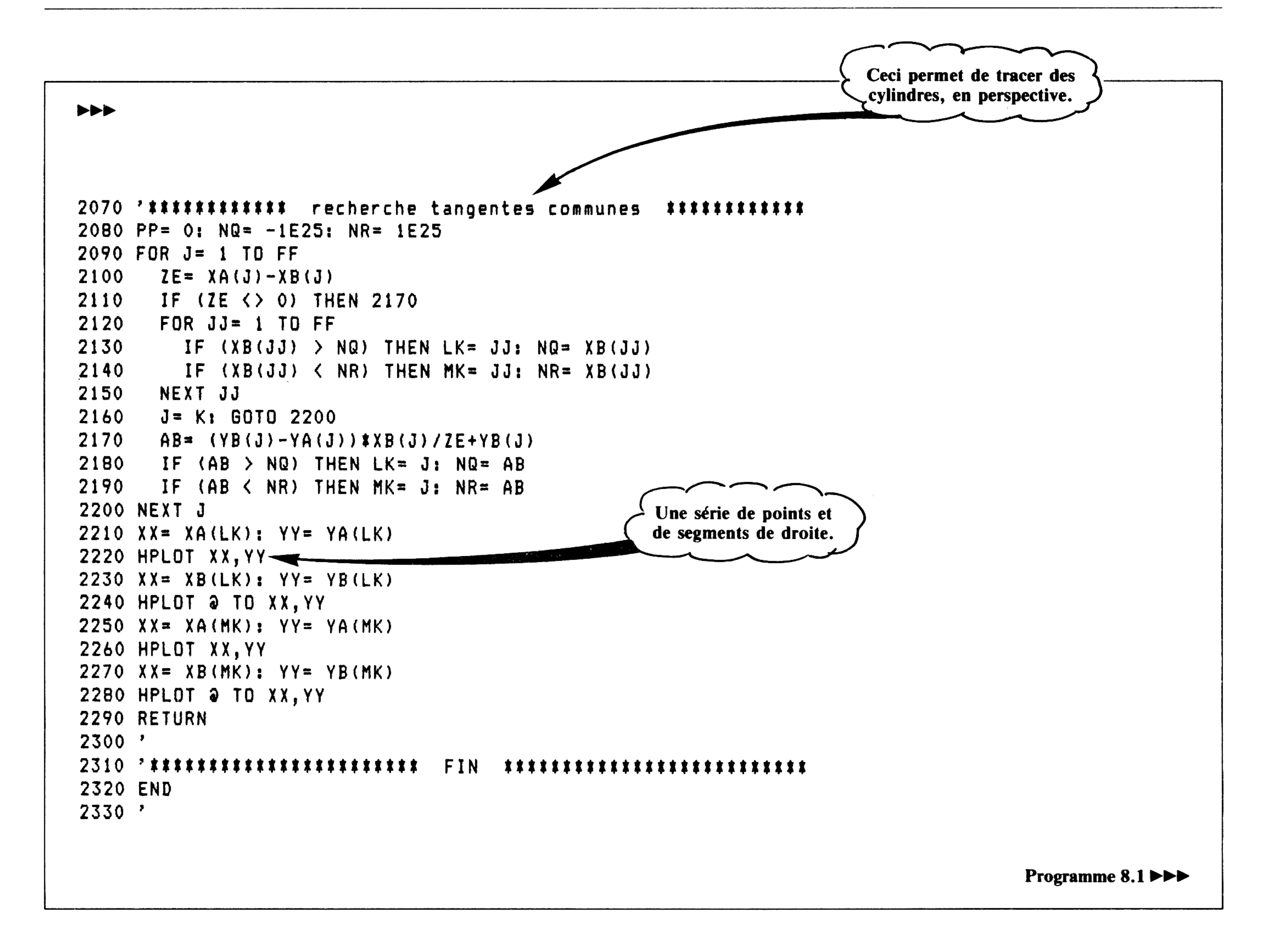
C'est très utile dans beaucoup de tracés.

```
1610 '************* trace des arcs et des cercles ***********
1620 DX= XX: DY= YY: DZ= ZZ
1630 ON RP GOTO 1660, 1800, 1940
1640 '
1650 '--------------- parallele au plan XY ----------------
1660 XX= DX+XR*COS(PT*JC): YY= DY+XR*SIN(PT*JC)
1670 K= 0: GOSUB 310: GOSUB 350
1680 HPLOT XF,YF
1690 FOR I= (PT+UV) TO QT STEP UV
1700   TI= I*JC
1710   XX= DX+XR*COS(TI): YY= DY+XR*SIN(TI)
1720   GOSUB 310: GOSUB 350
1730   HPLOT @ TO XF,YF
1740   IF (XY$ = "T") THEN K= K+1: XA(K)= XF: YA(K)= YF
1750 NEXT I
1760 IF (PP = 2) THEN GOSUB 2080
1770 RETURN
1780 '
1790 '--------------- parallele au plan YZ ----------------
1800 YY= DY+XR*COS(PT*JC): ZZ= DZ+XR*SIN(PT*JC)
1810 K= 0: GOSUB 310: GOSUB 350
1820 HPLOT XF,YF
1830 FOR I= (PT+UV) TO QT STEP UV
1840   TI= I*JC
1850   YY= DY+XR*COS(TI): ZZ= DZ+XR*SIN(TI)
1860   GOSUB 310: GOSUB 350
1870   HPLOT @ TO XF,YF
1880   IF (XY$ = "T") THEN K= K+1: XA(K)= XF: YA(K)= YF
1890 NEXT I
1900 IF (PP = 2) THEN GOSUB 2080
1910 RETURN
1920 '
1930 '--------------- parallele au plan XZ ----------------
1940 XX= DX+XR*COS(PT*JC): ZZ= DZ+XR*SIN(PT*JC)
1950 K= 0: GOSUB 310: GOSUB 350
1960 HPLOT XF,YF
1970 FOR I= (PT+UV) TO QT STEP UV
1980   TI= I*JC
1990   XX= DX+XR*COS(TI): ZZ= DZ+XR*SIN(TI)
2000   GOSUB 310: GOSUB 350
2010   HPLOT @ TO XF,YF
2020   IF (XY$ = "T") THEN K= K+1: XA(K)= XF: YA(K)= YF
2030 NEXT I
2040 IF (PP = 2) THEN GOSUB 2080
2050 RETURN
2060 '
```

Trace encore un point...

... puis une ligne.

Programme 8.1 ▶▶▶




```
▶▶▶

2070 '************  recherche tangentes communes  ************
2080 PP= 0: NQ= -1E25: NR= 1E25
2090 FOR J= 1 TO FF
2100   ZE= XA(J)-XB(J)
2110   IF (ZE <> 0) THEN 2170
2120   FOR JJ= 1 TO FF
2130     IF (XB(JJ) > NQ) THEN LK= JJ: NQ= XB(JJ)
2140     IF (XB(JJ) < NR) THEN MK= JJ: NR= XB(JJ)
2150   NEXT JJ
2160   J= K: GOTO 2200
2170   AB= (YB(J)-YA(J))*XB(J)/ZE+YB(J)
2180   IF (AB > NQ) THEN LK= J: NQ= AB
2190   IF (AB < NR) THEN MK= J: NR= AB
2200 NEXT J
2210 XX= XA(LK): YY= YA(LK)
2220 HPLOT XX,YY
2230 XX= XB(LK): YY= YB(LK)
2240 HPLOT @ TO XX,YY
2250 XX= XA(MK): YY= YA(MK)
2260 HPLOT XX,YY
2270 XX= XB(MK): YY= YB(MK)
2280 HPLOT @ TO XX,YY
2290 RETURN
2300 '
2310 '************************  FIN  ***************************
2320 END
2330 '
```

Programme 8.1 ▶▶▶

L'utilisation du logiciel est très simple, l'utilisateur se contentant de définir la *fenêtre* de projection à prendre en considération, les différentes couleurs et la position de l'observateur dans l'espace. Tout le restant (assemblages, projections, perspectives), est automatiquement généré par le micro-ordinateur. Le point le plus délicat consiste en la détermination des lignes de *DATA*, commençant à la ligne 2390. La marche à suivre est la suivante :

- tracer les différents plans composant l'objet, sur **papier millimétré**, comme indiqué sur la figure 8.1.

- puis relever les différentes coordonnées, en ayant toujours présent à l'esprit la situation du dessin où l'on se trouve, à tout instant.

Des exemples de tracés sont donnés aux figures 8.2 (un avion vu de côté, légèrement du dessus) et 8.3 (le même avion, dessiné à l'intérieur de quatre fenêtres différentes, vu du dessus, de l'avant, du dessous et de l'arrière).

En effet, la documentation des lignes de *DATA* se résume à indiquer au programme quelles sortes d'actions seront à générer, pour suivre les contours de l'objet à tracer, comme on le ferait avec la pointe d'un crayon qui parcourerait les lignes d'un objet *tridimensionnel.* Par exemple, pour effectuer un rapprochement avec le cas des tables traçantes, il est possible d'indiquer un positionnement « plume » levée ou un déplacement avec tracé « plume » baissée, pour des coordonnées identiques. Ainsi deux personnes différentes peuvent parvenir au même résultat, tout en empruntant des chemins différents, selon le point de départ, le sens choisi, l'arrêt éventuel ou non, etc. En voici les règles essentielles.

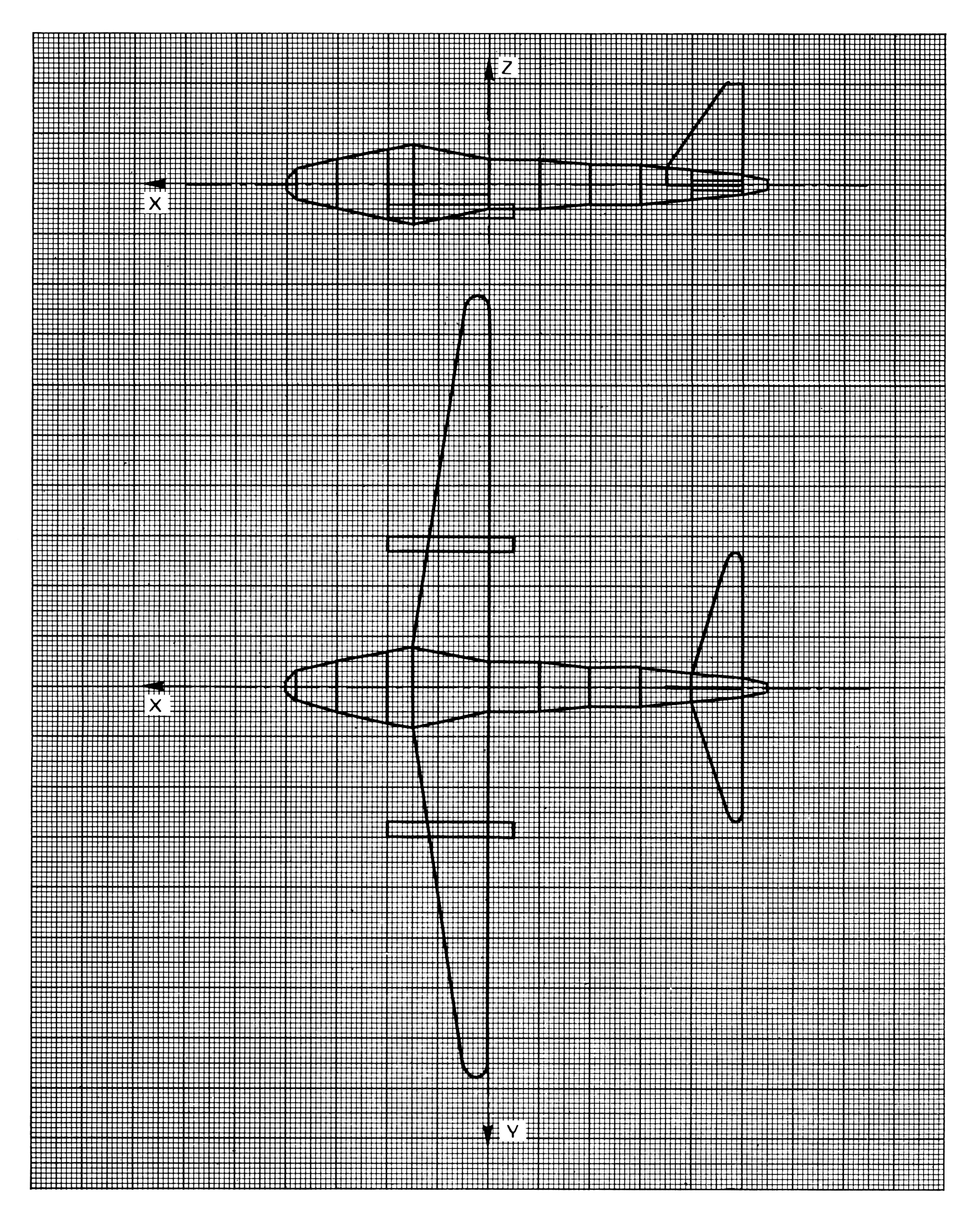


Figure 8.1 — Plans d'un avion, tracés sur papier millimétré, en vue de l'incorporation des cotes dans les lignes de DATA du programme.

```
▶▶▶
```

Début de la définition d'un objet à tracer. La numérotation peut ici commencer à 5000, par exemple.

```
2340 '==================================================
2350 '================  OBJET A TRACER  ===================
2360 '==================   A V I O N   ====================
2370 '
2380 'FUSELAGE
2390 DATA 45, 0, 0, 0
2400 DATA 40, 0, 0, 1
2410 DATA 40, 0, 0, 3, 1, 2
2420 DATA 38, 0, 0, 3, 3, 2
2425 DATA 38, 0, 0, 3, 3, 2
2430 DATA 30, 0, 0, 3, 5, 2
2435 DATA 30, 0, 0, 3, 5, 2
2440 DATA 20, 0, 0, 3, 7, 2
2445 DATA 20, 0, 0, 3, 7, 2
2450 DATA 15, 0, 0, 3, 8, 2
2455 DATA 15, 0, 0, 3, 8, 2
2460 DATA 0, 0, 0 ,3, 5, 2
2465 DATA 0, 0, 0 ,3, 5, 2
2470 DATA -10, 0, 0, 3, 5, 2
2475 DATA -10, 0, 0, 3, 5, 2
2480 DATA -20, 0, 0, 3, 4, 2
2485 DATA -20, 0, 0, 3, 4, 2
2490 DATA -30, 0, 0, 3, 4, 2
2495 DATA -30, 0, 0, 3, 4, 2
2500 DATA -42, 0, 0, 3, 3, 2
2505 DATA -42, 0, 0, 3, 3, 2
2510 DATA -50, 0, 0, 3, 2, 2
2515 DATA -50, 0, 0, 3, 2, 2
2520 DATA -55, 0, 0, 3, 1, 2
2530 '
2540 'AILE
2550 DATA 2.5, 75, -2, 2, 2.5, 1, 0, 180
2560 DATA 2.5, 75, -4, 2, 2.5, 1, 0, 180
2570 DATA 0, -75, -4, 1
2580 DATA 2.5, -75, -4, 2, 2.5, 1, 180, 360
2590 DATA 2.5, -75, -2, 2, 2.5, 1, 180, 360
2600 DATA 0, 75, -2, 0
2610 DATA 0, -75, -2, 1
2620 DATA 5, 75, -2, 0
2630 DATA 15, 8, -2, 1
2640 DATA 15, -8, -2, 1
2650 DATA 5, -75, -2, 1
2660 DATA 5, 75, -4, 0
2670 DATA 15, 8, -4, 1
2680 DATA 15, -8, -4, 1
2690 DATA 5, -75, -4, 1
2695 '
```

Le fuselage est une suite de cercles, avec leurs tangentes.

L'aile comporte des droites et des arcs de cercle.

```
2700 'AILERON
2710 DATA -48.5, 25, 1, 2, 1.5, 1, 0, 180
2720 DATA -48.5, 25, -1, 2, 1.5, 1, 0, 180
2730 DATA -50, -25, -1, 1
2740 DATA -48.5, -25, -1, 2, 1.5, 1, 180, 360
2750 DATA -48.5, -25, 1, 2, 1.5, 1, 180, 360
2760 DATA -50, 25, 1, 0
2770 DATA -50, -25, 1, 1
2780 DATA -47, 25, 1, 0
2790 DATA -42, 3, 1, 1
2800 DATA -42, -3, 1, 1
2810 DATA -47, -25, 1, 1
2820 DATA -47, 25, -1, 0
2830 DATA -42, 3, -1, 1
2840 DATA -42, -3, -1, 1
2850 DATA -47, -25, -1, 1
2860 '
2865 'REACTEURS
2870 DATA 20, 27.5, -6.5, 3, 2, 2
2880 DATA -5, 27.5, -6.5, 3, 2, 2
2890 DATA 20, -27.5, -6.5, 3, 2, 2
2900 DATA -5, -27.5, -6.5, 3, 2, 2
2910 '
2920 'DERIVE
2930 DATA -35, 0, 0, 0
2940 DATA -50, 0, 0, 1
2950 DATA -50, 0, 20, 1
2960 DATA -47, 0, 20, 1
2970 DATA -35, 0, 3, 1
2980 DATA -35, 0, 0, 1
2990 '
3000 'FIN
3010 DATA 0, 0, 0, 4
```

Fin des définitions de l'objet. A ne pas oublier !

Fin du programme 8.1

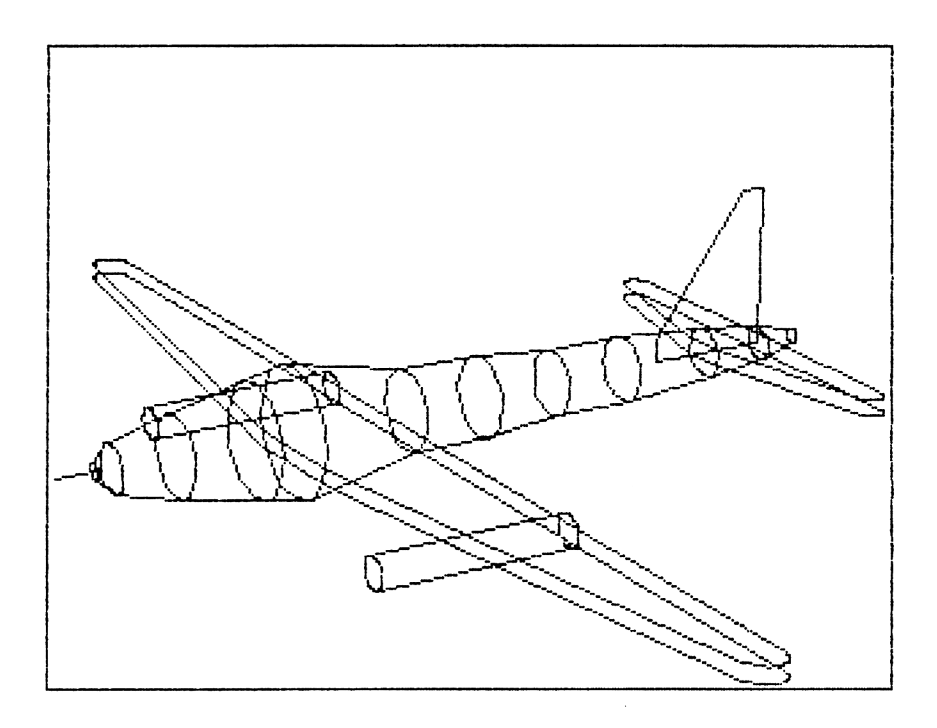


Figure 8.2 — Dessin de l'avion en trois dimensions et en perspective réelle, tracé par le programme.

Par exemple, à la ligne 2390, le logiciel commencera par dessiner le fuselage. Les trois premiers nombres concernent les coordonnées **x,y,z**, tandis que le quatrième concerne un code destiné à définir l'action à mener, en conséquence. Ainsi nous avons, pour cette ligne: **x=45**, **y=0**, **z=0** et **code=0**. La signification de ce code est la suivante:

Positionnement

0: indique un simple **positionnement** aux coordonnées qui précèdent, « plume » levée (donc, avec notre exemple, le tracé débute en positionnant la « plume » aux coordonnées **45,0,0**).

Traçage

1: dans ce cas, ce sera un déplacement **avec traçage** qui sera effectué, « plume » baissée. (A la ligne 2400, un trait sera donc dessiné, à partir des coordonnées **45,0,0** jusqu'à celles définies par l'ensemble **40,0,0**).

Cercle

3: les coordonnées représentent celles du **centre d'un cercle**. Il faut encore ajouter un cinquième nombre, fixant le **rayon** du cercle et un sixième, pouvant prendre les valeurs **1**, **2** ou **3**, selon que le cercle est *parallèle* respectivement aux plans **XY**, **YZ** ou **XZ**. (La ligne 2410 indique ainsi qu'un cercle devra être tracé, dont le centre sera situé aux coordonnées **40,0,0**, de rayon **1** et parallèle au plan **YZ**).

Arc de cercle

2: ce code est presque identique au précédent, mais il provoquera le dessin d'un simple **arc de cercle**, avec les mêmes significations que pour le code de type **3**, sauf en ce qui concerne le *point de départ* de l'arc. Celui-ci sera défini par un septième nombre, indiquant la valeur de l'**angle de départ** du tracé (en degrés) et par un huitième, fixant le **point d'arrivée** (représentant de même un angle en degrés).

Cette position dépend de la situation actuelle de la « plume ». En fixant l'angle de **0** à **180** degrés, l'arc sera dessiné dans la partie supérieure d'un cercle imaginaire, de la gauche vers la droite. Par contre, de **180** à **360** degrés, l'arc sera situé dans la partie basse de ce cercle, en partant de la droite vers la gauche. Par exemple, la ligne 2560 trace le bout arrondi inférieur de l'aile. L'arc de cercle sera défini par un centre de coordonnées **2,5**; **75** et **−4**, avec un rayon de **2,5**, parallèle au plan **XY**, débutant à **180** degrés et se terminant à **360** degrés, dans le sens horaire, par le bas).

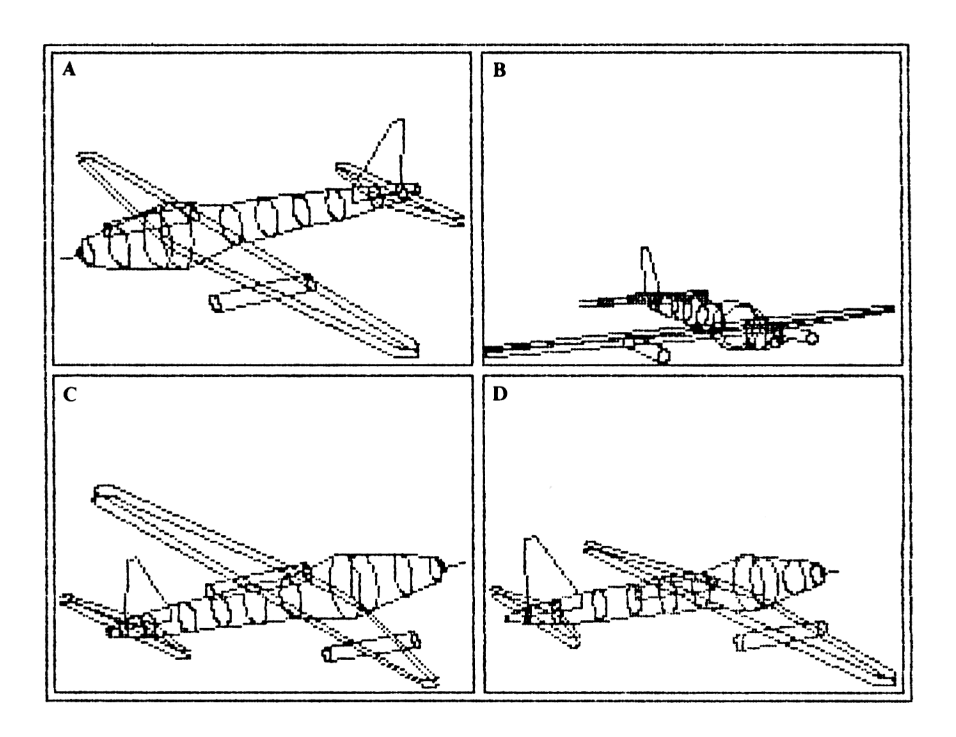


Figure 8.3 — Quatre représentations possibles de l'avion, en fonction du point de vue de l'observateur. En « A », vue du dessus. En « B », vue de l'avant. En « C », vue de dessous. En « D », vue de l'arrière.

Le point d'arrivée est conservé en mémoire, comme point de départ possible du tracé suivant, ce qui évite un repositionnement éventuel inutile. Ainsi à la ligne 2570, l'action indique un tracé « plume » baissée, du point d'arrivée de l'arc de la ligne 2560, jusqu'aux coordonnées **0**, **−75** et **−4**.

Tangentes

Les tangentes constituent un cas particulier de ces principes. Lorsqu'un cercle est tracé, ses coordonnées sont conservées en mémoire, en cas de besoin. Si la ligne de *DATA* suivante indique aussi qu'un cercle devra être tracé, un *top* sera positionné (« **T** »), afin de générer, par la suite, le tracé des tangentes à ces deux cercles. Le fuselage a été dessiné de cette façon. Ce procédé ne fonctionne que pour deux cercles consécutifs, afin de ne pas provoquer de générations parasites, non désirées au moment de l'élaboration de la *maquette.*

C'est ce qui explique la reprise en double des coordonnées définissant le fuselage, afin que les tangentes soient séquentielles. Sinon le dessin ne comporterait des tangentes qu'au niveau d'un cercle sur deux. Ainsi la ligne 2410 dessine un premier cercle. La ligne 2420 tracera un deuxième cercle, parallèle au premier, donc des tangentes communes seront automatiquement tracées. Mais afin de poursuivre le traçage de tangentes continuelles, représentant les pourtours du fuselage, il est nécessaire de définir un nouvel ensemble de deux cercles, débutant au niveau du dernier tracé. Donc de reprendre la définition du dernier cercle (ligne 2425, la ligne 2430 indiquant les caractéristiques du second cercle suivant).

4 signifie FIN

Enfin, il est absolument nécessaire d'indiquer au programme la fin générale du dessin, en attribuant au *code* le chiffre « **4** » (**x,y,z** étant de valeurs quelconques). La dernière ligne du programme (3010), effectue cette opération.

Ce programme, bien que n'effectuant pas l'élimination « automatique » des lignes cachées, est capable de dessiner n'importe quel objet, ou séries d'objets, selon les *lois* de la **troisième dimension** et de la **perspective**. Et cela de toutes les manières possibles et imaginables, comme nous le verrons par la suite. Les domaines d'applications sont immenses, variés et seulement limités par l'imagination.

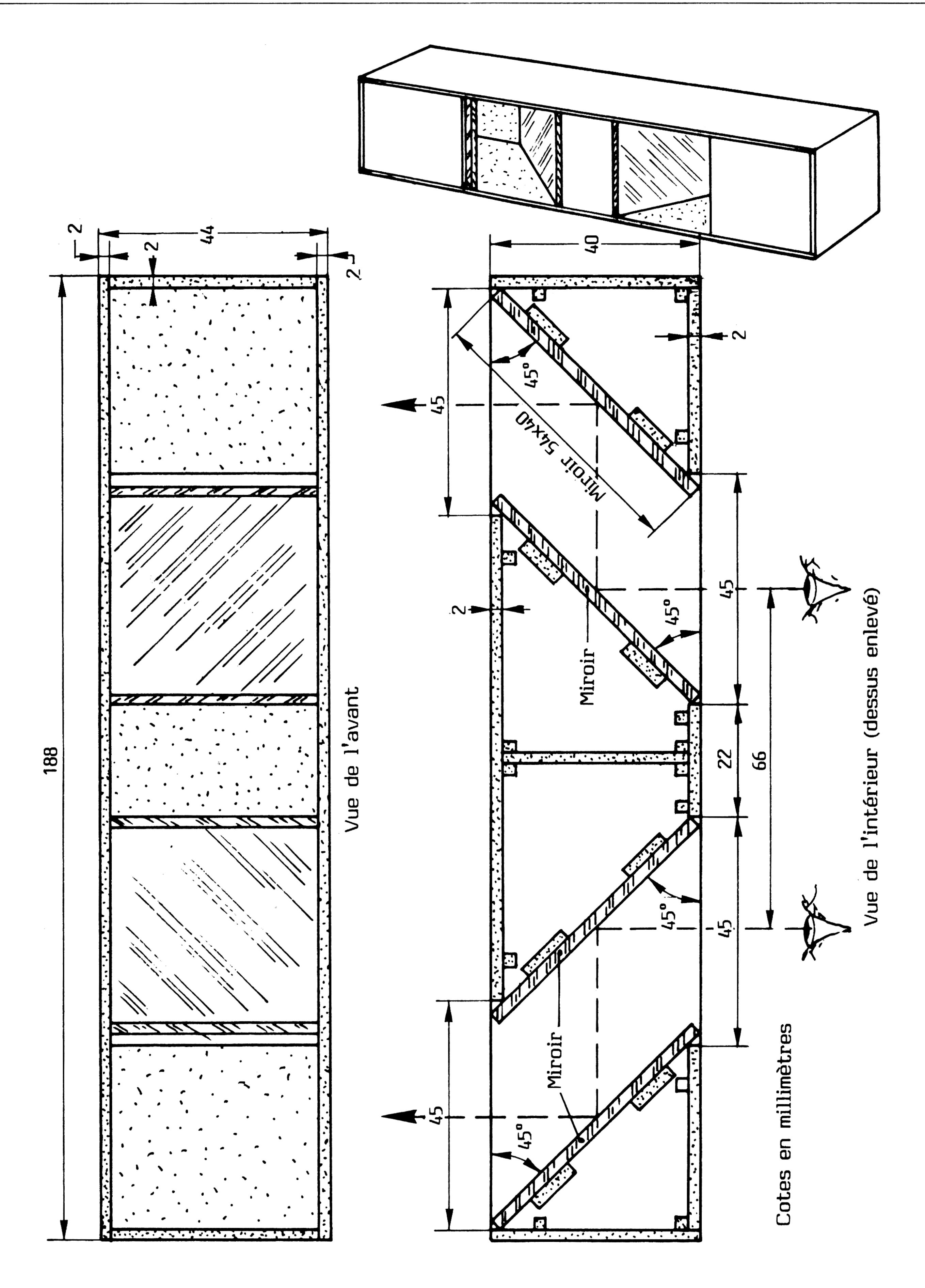


Figure 8.4 — Plans d'un stéréoscope simplifié, à quatre miroirs.

2. L'APPLICATION AU RELIEF

Dans le but de poursuivre les recherches ayant guidé les chapitres précédents, il peut être intéressant d'appliquer le relief aux images issues du programme ci-dessus. Bien que l'élimination des lignes cachées ne soit pas encore effectuée, la vision *stéréoscopique* peut cependant parfaitement être mise en œuvre. Le résultat obtenu ressemblera à un montage en « fils de fer », lequel n'est pas sans rappeler la représentation des fonctions en trois dimensions. Le principe restant le même, le cerveau continuera à être trompé et percevra le relief d'une façon parfaite. Tout ce qui a été dit précédemment, reste bien évidemment valable pour ce type de visualisation, mais un procédé supplémentaire de représentation *tridimensionnelle* peut être employé. Il est aussi possible d'augmenter la gamme des effets spéciaux dans ce domaine.

LA METHODE DES QUATRE MIROIRS

Vision stéréoscopique

Au niveau de la vision en *relief*, il est absolument nécessaire que chaque œil ne perçoive que l'image qui lui est destinée, afin que le cerveau puisse recréer la vision *stéréoscopique*. Si l'utilisateur ne dispose pas de lunettes spéciales, du type à verres colorés ou à filtres *polarisants*, il existe encore un moyen simple de construire un produit de remplacement. La figure 8.4 représente les plans d'un appareil permettant d'aboutir à un résultat satisfaisant.

Sa construction sera effectuée en contre-plaqué ou en « hobbystirène ». La seule difficulté consiste à se procurer quatre petits miroirs identiques. Il est possible d'acquérir de tels miroirs auprès des revendeurs de matériel automobile, sous la forme de « miroirs de courtoisie » autocollants. A l'aide d'un « diamant de vitrier », ils seront divisés en deux. Pour ce faire, il suffit de tracer un trait rayant le verre, à l'aide de l'outil, avec l'appui d'une règle, après avoir bien dégraissé la surface optique à découper. Puis insérez un petit clou, entre le miroir et la surface d'une table, par exemple, au niveau de la rayure et dans le même sens. En exerçant une simple pression sur les bords du miroir, ce dernier sera découpé instantanément en deux, selon une ligne de cassure franche et bien propre.

Dans le cas où ceux-ci seraient obtenus à partir d'une découpe, les cotes pourront être respectées. Par contre, s'ils proviennent de récupérations diverses, il conviendra d'essayer de s'en approcher le plus possible. L'assemblage effectué, les miroirs seront collés à leurs emplacements respectifs. Il est très important de veiller à leur inclinaison horizontale de **45** degrés, afin que le parallélisme concernant le trajet des rayons lumineux soit effectué d'une façon la plus parfaite possible.

A chaque œil son image

Puis le programme précédent sera lancé, de façon à obtenir soit deux petites images sur chaque moitié d'écran (c'est la solution ayant été retenue, pour les figures 8.5 et 8.6), soit deux fois une image recouvrant tout l'écran. En vue d'obtenir un résultat satisfaisant, il est pratique de bien noter les différents paramètres fixés lors du premier tracé. Lorsque le deuxième tracé est mis en œuvre, reprendre strictement ces nombres, en correspondance, sauf en ce qui concerne l'angle **THETA**. Celui-ci sera augmenté d'environ **six** degrés, pour la vue de droite.

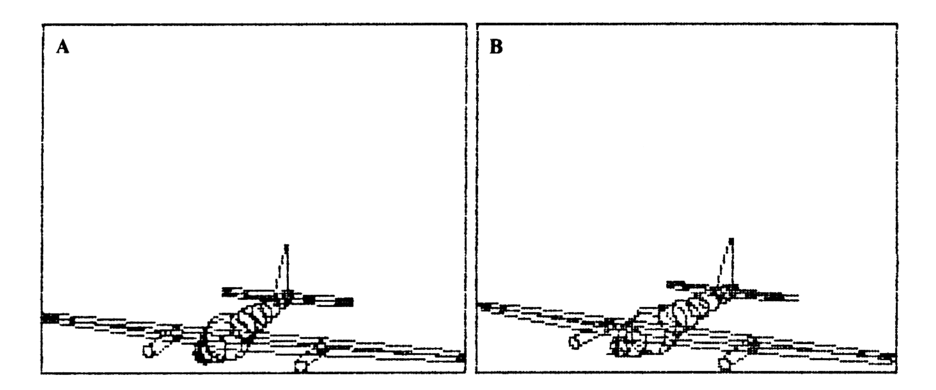


Figure 8.5 —Ensemble de deux images, destiné à percevoir la vue en relief d'un avion, par la méthode des miroirs. En A, vue de gauche. En B, vue de droite, dont l'angle THETA a été décalé de six degrés.

Relief sur écran ou sur papier

Avec ce procédé les couleurs n'ont aucune importance. La visualisation en relief pourra s'effectuer directement sur l'écran ou à l'aide d'une recopie d'écran imprimée sur papier. Pour ce faire, les deux images seront observées à l'aide de l'appareil décrit. Les miroirs font que chaque œil ne reçoit que l'image qui lui est destinée, leur double jeu annulant l'inversion indésirable des images. Dans le cas de la sortie imprimée, il est très facile de découper les deux images, dans le but de les centrer d'une façon optimale par rapport au « viseur ».

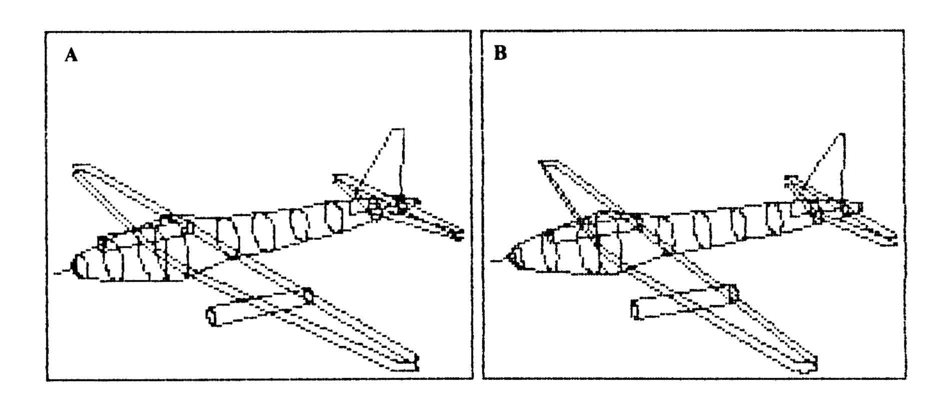


Figure 8.6 — Représentation identique de l'avion, en relief, vue sous un angle différent. En A, vue de gauche. En B, vue de droite.

Vision binoculaire

Il est intéressant de noter que l'écart de la *vision binoculaire*, en sortie du viseur, est supérieur à celui des yeux. Il s'ensuit que cet appareil est en fait un véritable **accentuateur** d'effet *stéréoscopique*. Il pourra être avantageusement utilisé pour augmenter la perception du relief, lors d'observations de paysages lointains, dans la réalité.

LE RELIEF SUR PLUSIEURS PLANS

Un autre procédé très intéressant est susceptible de générer des dessins se rapprochant davantage de la réalité, c'est-à-dire englobant plusieurs objets situés à des distances différentes. La figure 8.7 en montre un exemple possible. Six représentations d'avion ont été demandées, à divers emplacements de l'écran et à l'aide de *fenêtres* différentes. C'est ce qui permet d'agir sur la taille globale du

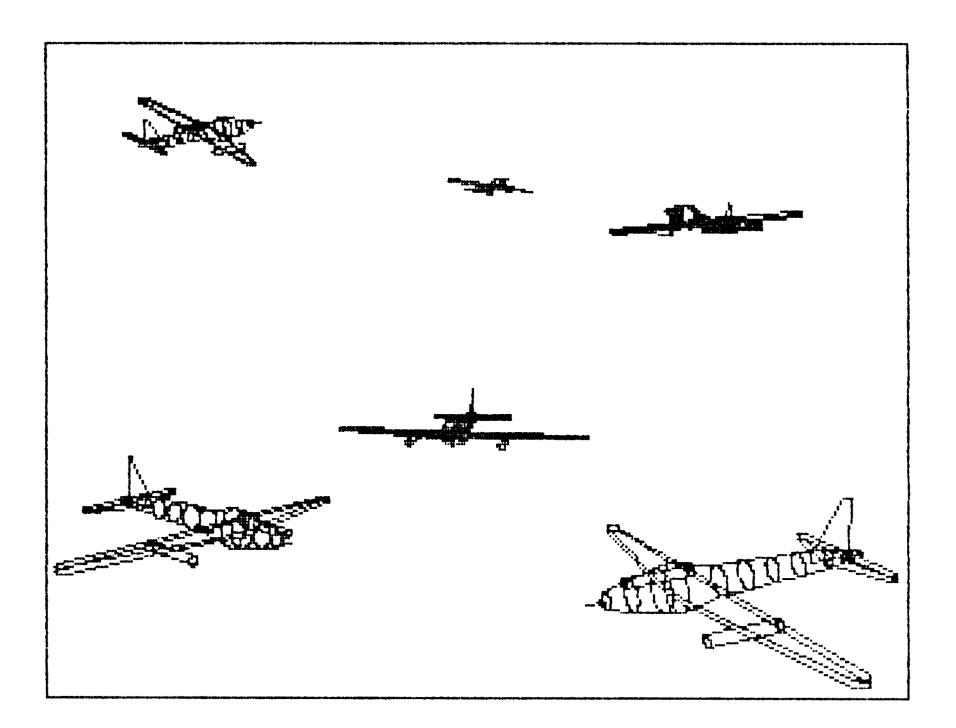


Figure 8.7 — Premier tracé d'une bataille aérienne, destiné à être visualisé en relief, par la méthode des miroirs. Vue de gauche.

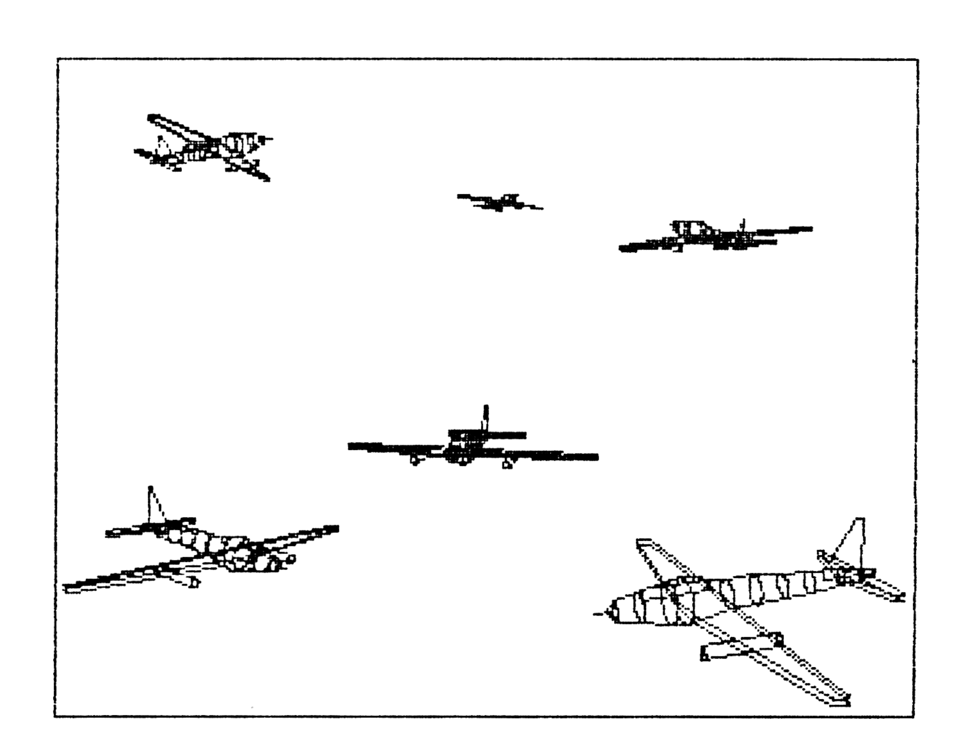


Figure 8.8 — Deuxième tracé d'une bataille aérienne, destiné à être visualisé en relief, par la méthode des miroirs. Vue de droite.

dessin. Le tracé le plus important, donc concernant un objet situé le plus près de l'observateur, s'est vu attribuer la valeur **300** pour **RHO**. Puis, en ordre décroissant, les valeurs suivantes ont été données à **RHO**: **350**, **400**, **425**, **450** et **475**.

Tous les paramètres entrés sont toujours soigneusement notés. Cette première représentation de six objets peut être tracée à l'aide de la couleur bleue, dans le cas d'une visualisation à l'aide des lunettes à verres colorés. Elle concernera la vue de **gauche**.

L'angle THETA modifie la vue de droite

Pour réaliser la vue de **droite** (Figure 8.8), les mêmes paramètres seront réintroduits, sauf en ce qui concerne la couleur (rouge) et les différentes valeurs des angles **THETA**. Mais à présent ces nombres devront être modifiés en fonction de l'éloignement relatif de l'objet par rapport à l'observateur. Dans le cas de l'avion le plus proche (**RHO = 300**), la différence d'angle a été fixée à **+6** degrés. Puis, en fonction de l'éloignement, **+5** degrés, **+4** degrés, **+3** degrés, **+2** degrés et **+1** degré. Car plus un objet est rapproché, plus la différence de l'angle de vision par rapport aux deux yeux est grande.

En vue d'appréhender le relief, l'utilisateur pourra reprendre les méthodes développées ci-avant. Par exemple l'utilisation des lunettes à verres colorés, bleu à gauche et rouge pour la droite. Dans ce cas le clignotement habituel entre les deux pages d'écran sera produit par le court programme suivant, aisément adaptable en fonction du matériel utilisé :

```
10 FOR I = 1 TO 32000 : HPAGE 0 : HPAGE 1 : NEXT I
20 GOTO 10
```

En visualisation *stéréoscopique*, l'effet rendu sera saisissant. L'utilisateur aura vraiment l'impression de se situer au milieu d'une bataille aérienne, avec des avions situés près de lui et d'autres à des distances variables. Il ne faut pas hésiter à expérimenter largement dans ce domaine, à l'aide de plusieurs types de représentations tridimensionnelles (photographies notamment) et même en dessinant de nombreux objets variés, paramétrés de diverses manières.

Une vraie bataille aérienne !

3. LES RETOUCHES ET LA D.A.O.

Il est difficile et surtout très pénalisant en espace mémoire ainsi qu'en temps d'exécution, de provoquer l'effacement automatique des surfaces cachées, dans le cas de tracés d'objets de formes très diverses. Or il existe un moyen de s'affranchir de ce genre de limitation. De même que le photographe améliore ses photos en les retouchant, il est fortement conseillé de procurer aux représentations graphiques une finition susceptible d'en améliorer énormément la représentation artistique. Dans cette optique, il conviendra d'utiliser avec profit les différents logiciels graphiques fournis avec chaque micro-ordinateur.

Selon la sophistication du logiciel associé, les flèches du clavier pourront être utilisées pour des tracés de couleurs différentes ou avec la couleur du fond, pour « gommer ». Une « souris » permettra une manipulation plus agréable et plus rapide. Le cas échéant, le lecteur pourra même créer son propre logiciel de retouches, en assembleur ou en langage compilé.

De la C.A.O. à la D.A.O.

Dans un premier temps, les lignes cachées seront « gommées », ce qui conduira à une représentation graphique plus proche de la réalité. Puis des retouches proprement dites pourront être effectuées. C'est-à-dire que l'on peut considérer que le programme a permis de tracer l'*ossature* de l'objet, en respectant les lois de la visualisation dans l'espace tridimensionnel. Puis l'utilisateur a alors la possibilité « d'habiller » ses dessins de façon beaucoup plus artistique, en rectifiant une ligne par exemple, en ajoutant des courbures et autres détails, en employant des couleurs différentes lors de remplissages de surfaces etc. Le résultat obtenu peut être très réussi, l'effet de perspective étant garanti par l'utilisation du logiciel. Avec beaucoup de soin et de la patience, la **C.A.O.** peut même devenir de la véritable **D.A.O.**

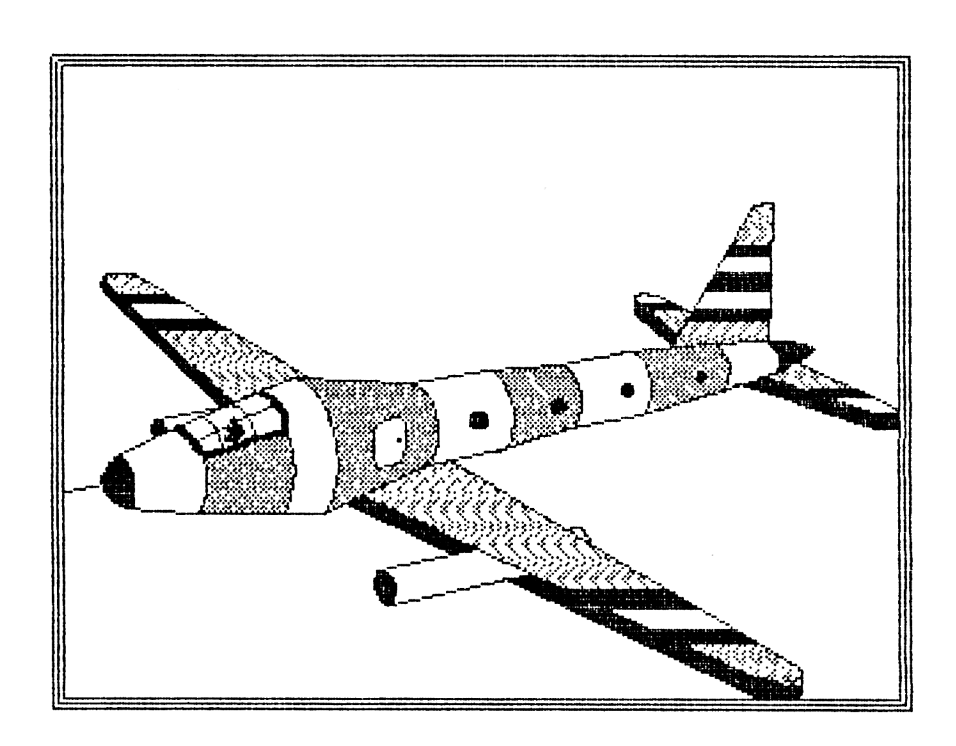


Figure 8.9 — Essai de D.A.O., à partir d'un dessin d'avion, généré par programme.

Le relief en plus

Un essai (modeste) allant dans ce sens, a été tenté avec la figure 8.9. Le résultat peut largement être amélioré, mais il montre ce qu'il est possible d'effectuer à l'aide de ce procédé, dont les seules limites sont celles de l'imagination. Ne pas oublier non plus les possibilités d'exploitation au niveau du **relief**, en respectant les principes désormais bien assimilés.

4. LES ANIMATIONS

Deux pages écran

Lorsque le logiciel graphique permet la visualisation de deux pages d'écran en haute résolution, des effets intéressants d'animation peuvent être obtenus, en temps réel, même si les objets montrés comportent un grand nombre de détails. Le principe retenu consiste à générer le dessin d'un objet sur une page d'écran. Puis à générer une deuxième représentation de cet objet sur la deuxième page, mais avec quelques modifications sur le plan de la situation spatiale de différentes parties constituant cet objet. En provoquant l'affichage périodique alterné des deux pages d'écran, avec des temporisations adéquates, l'effet d'animation sera très bien rendu.



Il est de même possible de concevoir un petit logiciel en *langage machine*, en langage *compilé* ou même en *BASIC* amélioré, permettant de recopier les zones d'écran en mémoire centrale et vice-versa, dans le but de simuler l'existence de deux pages écrans. De cette façon il serait possible de provoquer un affichage alterné des deux zones mémoire représentant les dessins, ce qui aurait pour résultat de simuler l'existence de deux pages écrans en haute résolution graphique, de rendre possible des effets d'animation et de pouvoir afficher des images en vrai relief directement sur un écran vidéo.

La figure 8.10 reprend l'exemple de l'avion, mais la figure 8.11 a introduit quelques modifications relatives au positionnement des ailes. Ces dernières ont été tracées avec leur extrêmité respective située un peu plus haut (en modifiant les lignes de *DATA* du programme 8.1 par remplacement des lignes concernées, répertoriées dans le programme 8.2). En affichant alternativement les pages graphiques à l'aide du programme suivant, par exemple, l'utilisateur verra son avion **battre des ailes**.

10 HPAGE 0 : FOR I= 1 TO 500 : NEXT I
20 HPAGE 1 : FOR I= 1 TO 500 : NEXT I
30 GOTO 10

Attachez vos ceintures !

Programme 8.2.

```
2530 '
2540 'AILE
2550 DATA 2.5,  75,  8,  2,  2.5,  1,   0, 180
2560 DATA 2.5,  75,  6,  2,  2.5,  1,   0, 180
2570 DATA   0,   5, -4,  1
2572 DATA   0,  -5, -4,  1
2574 DATA   0, -75,  6,  1
2580 DATA 2.5, -75,  6,  2,  2.5,  1, 180, 360
2590 DATA 2.5, -75,  8,  2,  2.5,  1, 180, 360
2600 DATA   0, -75,  8,  0
2610 DATA   0,  -5, -2,  1
2612 DATA   0,   5, -2,  1
2614 DATA   0,  75,  8,  1
2620 DATA   5,  75,  8,  0
2630 DATA  15,   8, -2,  1
2640 DATA  15,  -8, -2,  1
2650 DATA   5, -75,  8,  1
2660 DATA   5,  75,  6,  0
2670 DATA  15,   8, -4,  1
2680 DATA  15,  -8, -4,  1
2690 DATA   5, -75,  6,  1
2695 '
2865 'REACTEURS
2870 DATA  20,  27.5,  0, 3, 2, 2
2880 DATA  -5,  27.5,  0, 3, 2, 2
2890 DATA  20, -27.5,  0, 3, 2, 2
2900 DATA  -5, -27.5,  0, 3, 2, 2
2910 '
```

Peu de modifications, mais beaucoup de conséquences...

Fin des modifications

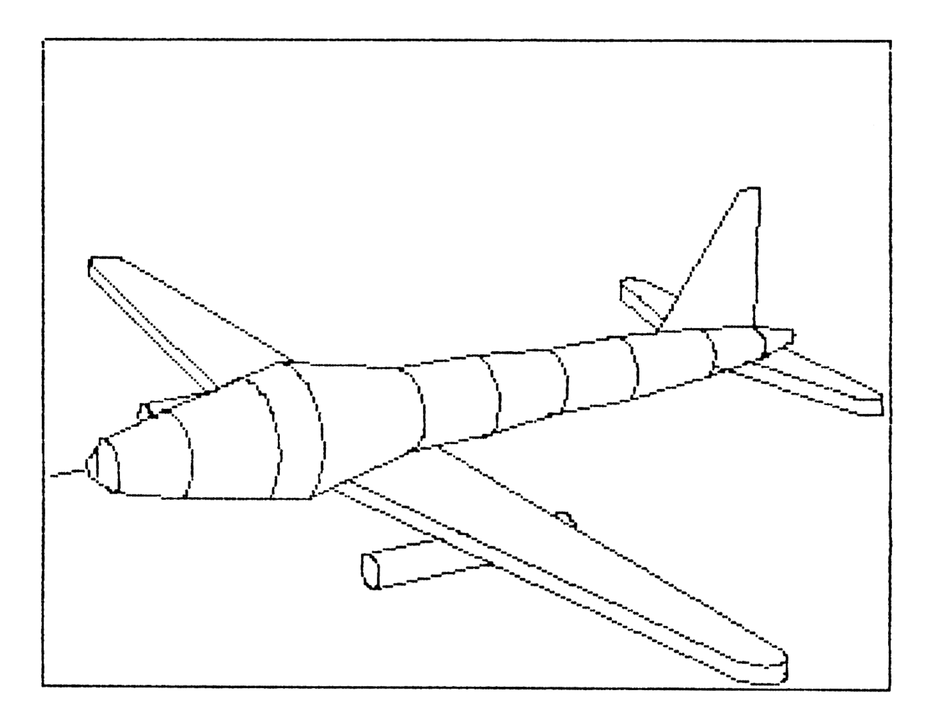


Figure 8.10 — Première recopie d'écran d'une animation générée en temps réel. Première position des ailes de l'avion.

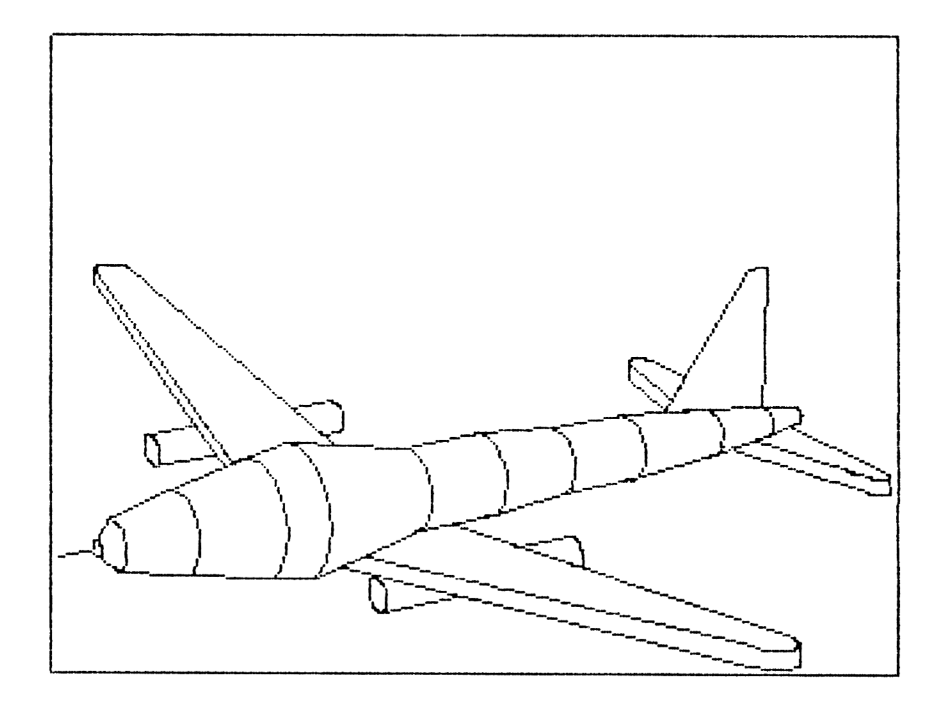


Figure 8.11 — Deuxième recopie d'écran d'une animation générée en temps réel. Deuxiéme position des ailes de l'avion. L'ensemble des deux vues provoque un battement d'ailes de l'avion.

Dans le domaine de l'aéronautique, gageons que ce mode de vol est une réalisation d'importance capitale, suggérée à l'entreprise « DASSAULT »...

Voici maintenant terminée une première série de réalisations graphiques, permettant de familiariser le lecteur avec le programme de **C.A.O** précédent. Par contre ces applications ne sont pas limitatives. Leur but est de stimuler simplement l'imagination de l'utilisateur, en vue de favoriser le développement de réalisations intéressantes. Il suffit parfois d'un simple petit détail, permettant la combinaison de plusieurs procédés, pour obtenir des résultats peut-être spectaculaires.

CHAPITRE 9

QUELQUES APPLICATIONS

Beaucoup de domaines peuvent être concernés par les techniques graphiques étudiées jusqu'à présent. Certaines applications utiliseront davantage les techniques relatives au *relief*, tandis que d'autres relèveront principalement du domaine de la *troisième dimension.* Afin de mieux montrer les résultats qu'il est possible d'obtenir dès à présent, quelques réalisations vont être développées.

Quatre domaines de la micro-informatique, particulièrement sensibles, seront abordés: celui de l'**enseignement**, au sein duquel l'utilisation du *relief* sera prépondérante; celui de la **mécanique**, privilégiant la *troisième dimension* (**C.A.O.**); celui de la **ludotique**, apportant les principes du dessin (**D.A.O**) aux jeux et enfin celui de la **gestion**, par exemple dans le cas des *histogrammes* performants.

1. LA GEOMETRIE DANS L'ESPACE

L'espace à trois dimensions

S'il est un domaine dans lequel l'apport du **relief** n'est vraiment pas un « gadget », c'est bien celui de la géométrie dans l'espace. Chacun de nous a en effet éprouvé les mêmes difficultés à essayer d'imaginer un espace *tridimensionnel*, pauvrement suggéré par de vagues tracés sur une feuille de papier. Lorsque les dessins géométriques n'atteignent pas une complexité trop élevée, la puissance imaginative du cerveau est capable de concevoir une représentation réelle des divers plans, droites, angles et volumes divers. Mais dès qu'elle atteint un certain niveau, beaucoup de personnes peuvent éprouver de sérieuses difficultés à résoudre leurs problèmes.

La géométrie dans l'espace... mais c'est très simple !

Maintenant, grâce à l'emploi devenu réalisable du relief, ces contraintes sont supprimées. En effet, il est vraiment très facile de faire générer par l'ordinateur tout problème de géométrie dans l'espace, en lui indiquant simplement les coordonnées tridimensionnelles des différentes figures géométriques. Le logiciel habituel se chargera alors d'en assurer la représentation **réelle** dans l'espace, sous n'importe quel angle de vision, selon les désidérata de l'utilisateur. Un ensemble de rotations diverses pourra être aisément opéré et la visualisation en relief permettra d'en percevoir les différentes composantes, comme si elles étaient **réellement** suspendues dans l'espace.

Pour parvenir à ces résultats, il suffit de reprendre le programme 8.1 et de remplacer le contenu des lignes de *DATA*, en fonction de ce que l'on veut obtenir. Pour illustrer ce principe, la figure 9.1 représente un ensemble de deux plans se coupant dans l'espace, deux droites quelconques, un cylindre et une pyramide. On ne peut vraiment pas affirmer que la représentation sur papier soit évidente.

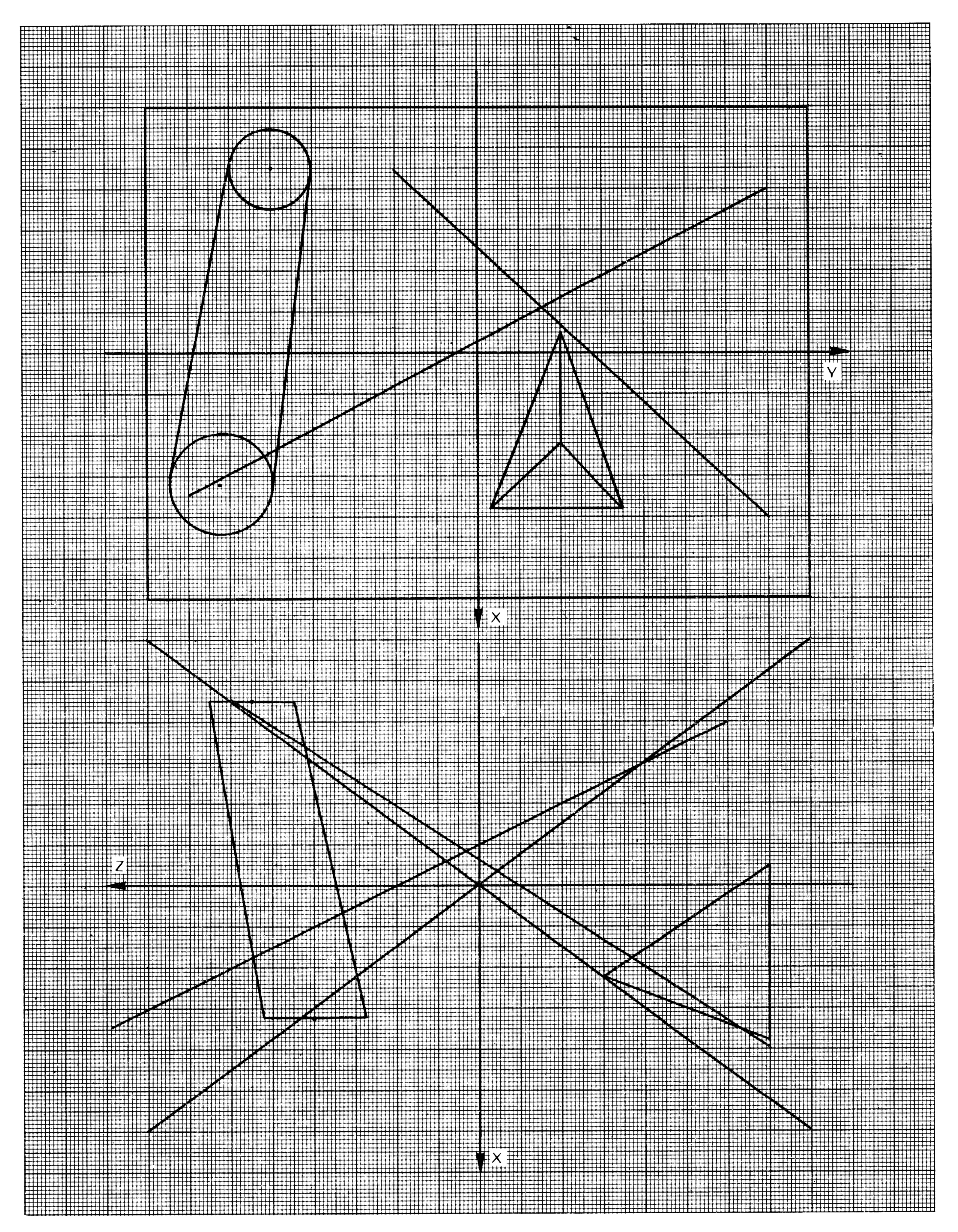


Figure 9.1 — Ensemble de plans, droites, cylindre et pyramide, représentés sur papier millimétré.

4 signifie FIN

Les différentes coordonnées composant ces figures seront donc reprises dans les lignes de *DATA* suivantes (2360 à 2750), remplaçant celles du programme précité. Elles concernent chacun des deux plans, leur intersection, les deux droites, le cylindre et la pyramide. Il ne faut surtout pas oublier la dernière ligne (2750), indiquant la fin de l'ensemble des *DATA*.

A recopier à la fin du programme précédent.

```
2360 '================ G E O M E T R I E ==================
2370 '
2380 '1er PLAN, HORIZ.
2390 DATA  60, -80,  80, 0
2400 DATA -60, -80, -80, 1
2410 DATA -60,  80, -80, 1
2420 DATA  60,  80,  80, 1
2430 DATA  60, -80,  80, 1
2440 '
2450 '2eme PLAN, VERT.
2460 DATA  60, -80, -80, 0
2470 DATA -60, -80,  80, 1
2480 DATA -60,  80,  80, 1
2490 DATA  60,  80, -80, 1
2500 DATA  60, -80, -80, 1
2510 '
2512 'INTERSECTION
2514 DATA   0, -80,   0, 0
2516 DATA   0,  80,   0, 1
2518 '
2520 '1ere DROITE, DROITE
2530 DATA  40,  70, -70, 0
2540 DATA -45, -20,  60, 1
2550 '
2560 '2eme DROITE, GAUCHE
2570 DATA -40,  70, -60, 0
2580 DATA  35, -70,  90, 1
2590 '
2600 'CYLINDRE
2610 DATA  32, -63,  40, 3, 12, 2
2620 DATA -45, -50,  55, 3, 10, 2
2630 '
2640 'PYRAMIDE
2650 DATA  38,   3, -70, 0
2660 DATA  38,  35, -70, 1
2670 DATA  -5,  20, -70, 1
2680 DATA  38,   3, -70, 1
2690 DATA  22,  20, -30, 1
2700 DATA  38,  35, -70, 1
2710 DATA  22,  20, -30, 0
2720 DATA  -5,  20, -70, 1
2730 '
2740 'FIN
2750 DATA   0,   0,   0, 4
```

Seules les droites sont à tracer dans ce cas.

Traçage de deux cercles et des tangentes qui les relient.

A ne pas oublier, indique la fin des données !

Listage 9.1

Paramétrage

En respectant les principes habituels, la figure 9.2 a été obtenue en choisissant les paramètres suivants, lors du lancement du programme :

RHO=400

couleur **bleue**

THETA=10

PHI=10



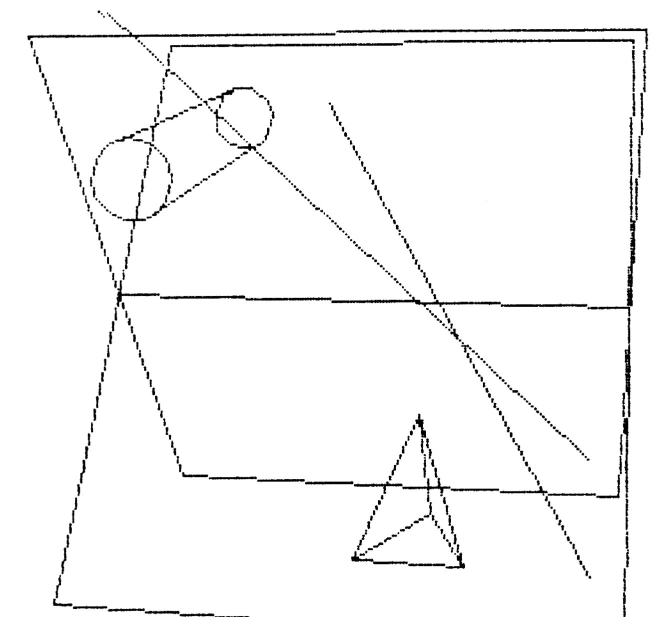


Figure 9.2 — Première représentation de plans, droites, cylindre et pyramide, en trois dimensions et vue réelle. Pour le relief, image de gauche.

Le résultat obtenu serait celui tracé par tout étudiant en *géométrie spatiale*, avec toutefois l'apport d'une perspective « réelle ». En vue d'en obtenir la représentation en **relief**, il a été nécessaire de générer une deuxième vue sur la seconde page d'écran, en reprenant strictement les mêmes paramètres, sauf pour **THETA** qui a été fixé à **20** et la couleur employée étant le **rouge**. Le résultat est représenté par la figure 93.

Le miracle du relief !

A ce moment, toutes les techniques étudiées jusqu'à présent peuvent être employées, dans le but de percevoir le **relief** de ces figures géométriques. C'est-à-dire soit en faisant clignoter les deux pages écran et en observant le résultat avec les lunettes bicolores, soit en visualisant la reproduction sur papier à l'aide du viseur à quatre miroirs (figure 8.4) ou bien encore en photographiant les

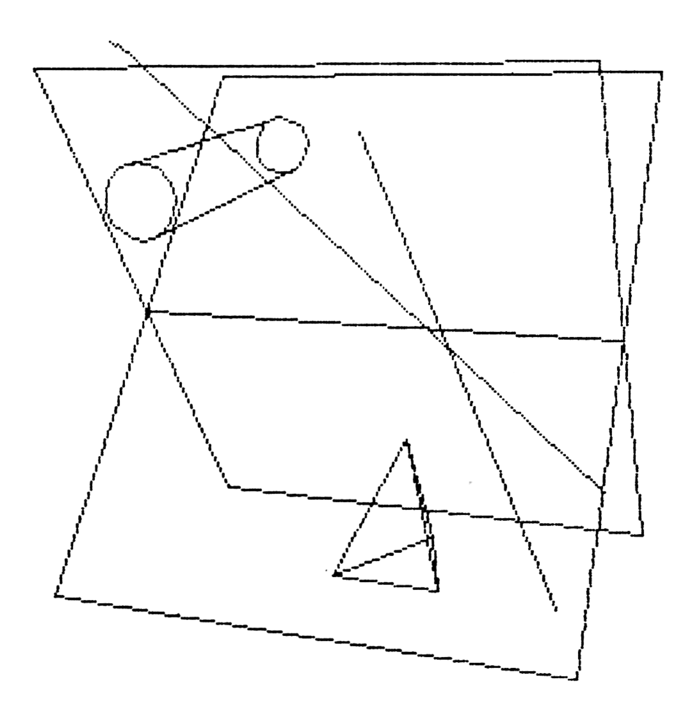


Figure 9.3 — Deuxième représentation de plans, droites, cylindre et pyramide, en trois dimensions et vue réelle. Pour le relief, image de droite.

écrans selon les techniques étudiées ci-avant. Désormais, un étudiant utilisant ce procédé n'aura plus aucune excuse de ramener une mauvaise note en géométrie dans l'espace...

L'auteur accepte les chèques...

2. LA MECANIQUE

Empilement séquentiel

Parmi les branches industrielles qui utilisent le plus la **C.A.O.**, la mécanique parait bien en occuper la première place. Il faut reconnaître qu'il s'agit d'une méthode idéale, pour concevoir des systèmes complexes. Afin de mieux en démontrer l'utilité, nous allons appliquer le programme générateur de dessins *tridimensionnels* à la représentation en « éclaté » d'une pièce mécanique indispensable à tout véhicule: la fixation supérieure d'un élément de suspension. Il s'agit d'un cas typique, montrant bien l'empilement séquentiel de plusieurs pièces de formes différentes (écrous, rondelles, coupelles, etc).

La figure 9.4 en montre le plan simplifié, tracé sur papier millimétré, selon la méthode habituelle. Les différentes coordonnées représentant chacune de ces pièces sont ensuite entrées en lignes de *DATA*, comme ci-dessous. L'axe de symétrie passant par le centre géométrique de toutes les pièces, il est commode de prendre la valeur « **0** » pour la coordonnée « **Z** ».

Déplacements autour de la pièce mécanique

Après avoir remplacé les lignes de *DATA* du programme 8.1 par ces dernières, le résultat obtenu est conforme à celui de la figure 9.5. La juxtaposition des différentes pièces parait évidente. Naturellement, en reprenant les techniques énoncées ci-avant, il est possible à l'aide d'un logiciel graphique approprié, de « gommer » les lignes cachées et d'améliorer le dessin. De cette façon le travail principal ne consiste qu'en des retouches, tandis que le programme en a effectué la tâche la plus ardue, à savoir représenter l'ensemble, dans l'espace, en **trois dimensions réelles**. Et cela quelle que soit la situation spatiale de l'observateur. La figure 9.6 en montre une représentation possible.

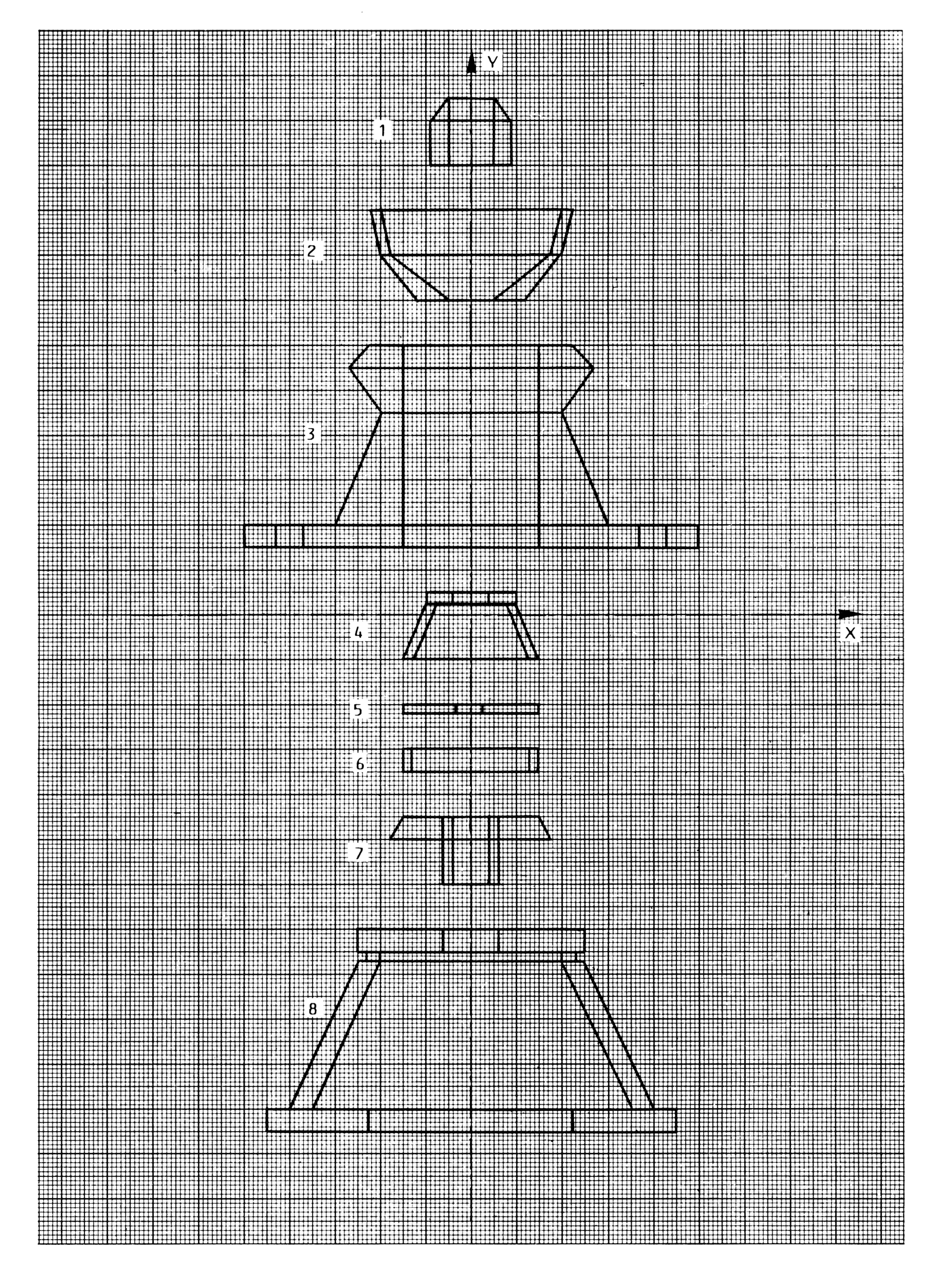


Figure 9.4 — Plan d'une pièce mécanique, tracé sur papier millimétré.

Remplace les lignes de « data » du programme précédent.

```
2360 '================ M E C A N I Q U E =============
2370 '
2380 'PIECE 1
2390 DATA   0, 115,   0, 3,  4, 3
2400 DATA   0, 100,   0, 3,  4, 3
2410 DATA   0, 115,   0, 3,  5, 3
2420 DATA   0, 110,   0, 3,  9, 3
2430 DATA   0, 110,   0, 3,  9, 3
2440 DATA   0, 100,   0, 3,  9, 3
2450 '
2460 'PIECE 2
2470 DATA   0,  90,   0, 3, 20, 3
2480 DATA   0,  80,   0, 3, 18, 3
2490 DATA   0,  80,   0, 3, 18, 3
2500 DATA   0,  70,   0, 3,  4, 3
2510 DATA   0,  90,   0, 3, 22, 3
2520 DATA   0,  80,   0, 3, 20, 3
2530 DATA   0,  80,   0, 3, 20, 3
2540 DATA   0,  70,   0, 3, 12, 3
2550 '
2560 'PIECE 3
2570 DATA   0,  60,   0, 3, 22, 3
2580 DATA   0,  55,   0, 3, 27, 3
2590 DATA   0,  55,   0, 3, 27, 3
2600 DATA   0,  45,   0, 3, 20, 3
2610 DATA   0,  45,   0, 3, 20, 3
2620 DATA   0,  20,   0, 3, 50, 3
2630 DATA   0,  20,   0, 3, 50, 3
2640 DATA   0,  15,   0, 3, 50, 3
2650 DATA   0,  60,   0, 3, 15, 3
2660 DATA   0,  15,   0, 3, 15, 3
2670 DATA  40,  20,   0, 3,  3, 3
2680 DATA  40,  15,   0, 3,  3, 3
2690 DATA -40,  20,   0, 3,  3, 3
2700 DATA -40,  15,   0, 3,  3, 3
2710 '
2720 'PIECE 4
2730 DATA   0,   5,   0, 3, 10, 3
2740 DATA   0,   3,   0, 3, 10, 3
2750 DATA   0,   3,   0, 3, 10, 3
2760 DATA   0, -10,   0, 3, 15, 3
2770 DATA   0,   5,   0, 3,  4, 3
2780 DATA   0,   3,   0, 3,  4, 3
2790 DATA   0,   3,   0, 3,  8, 3
2800 DATA   0, -10,   0, 3, 13, 3
2870 '
```

Ici, ce ne seront que des cercles à tracer, avec leurs tangentes.

Il vaut mieux repérer les différentes pièces, en cas de modification ultérieure...

```
2880 'PIECE 5
2890 DATA   0, -20,   0, 3,  4, 3
2900 DATA   0, -22,   0, 3,  4, 3
2910 DATA   0, -20,   0, 3, 15, 3
2920 DATA   0, -22,   0, 3, 15, 3
2930 '
2940 'PIECE 6
2950 DATA   0, -30,   0, 3, 13, 3
2960 DATA   0, -35,   0, 3, 13, 3
2970 DATA   0, -30,   0, 3, 15, 3
2980 DATA   0, -35,   0, 3, 15, 3
2990 '
3000 'PIECE 7
3010 DATA   0, -45,   0, 3,  4, 3
3020 DATA   0, -60,   0, 3,  4, 3
3030 DATA   0, -45,   0, 3,  6, 3
3040 DATA   0, -60,   0, 3,  6, 3
3050 DATA   0, -45,   0, 3, 15, 3
3060 DATA   0, -50,   0, 3, 18, 3
3070 '
3080 'PIECE 8
3090 DATA   0, -70,   0, 3,  6, 3
3100 DATA   0, -75,   0, 3,  6, 3
3110 DATA   0, -70,   0, 3, 25, 3
3120 DATA   0, -75,   0, 3, 25, 3
3130 DATA   0, -75,   0, 3, 20, 3
3140 DATA   0, -77,   0, 3, 20, 3
3150 DATA   0, -75,   0, 3, 23, 3
3160 DATA   0, -77,   0, 3, 23, 3
3170 DATA   0, -77,   0, 3, 20, 3
3180 DATA   0,-110,   0, 3, 35, 3
3190 DATA   0, -77,   0, 3, 25, 3
3200 DATA   0,-110,   0, 3, 40, 3
3210 DATA   0,-110,   0, 3, 22, 3
3220 DATA   0,-115,   0, 3, 22, 3
3230 DATA   0,-110,   0, 3, 45, 3
3240 DATA   0,-115,   0, 3, 45, 3
3250 '
3260 'FIN
3270 DATA   0, 0, 0, 4
```

Et voici la fin des données, impérative.

Listage 9.2

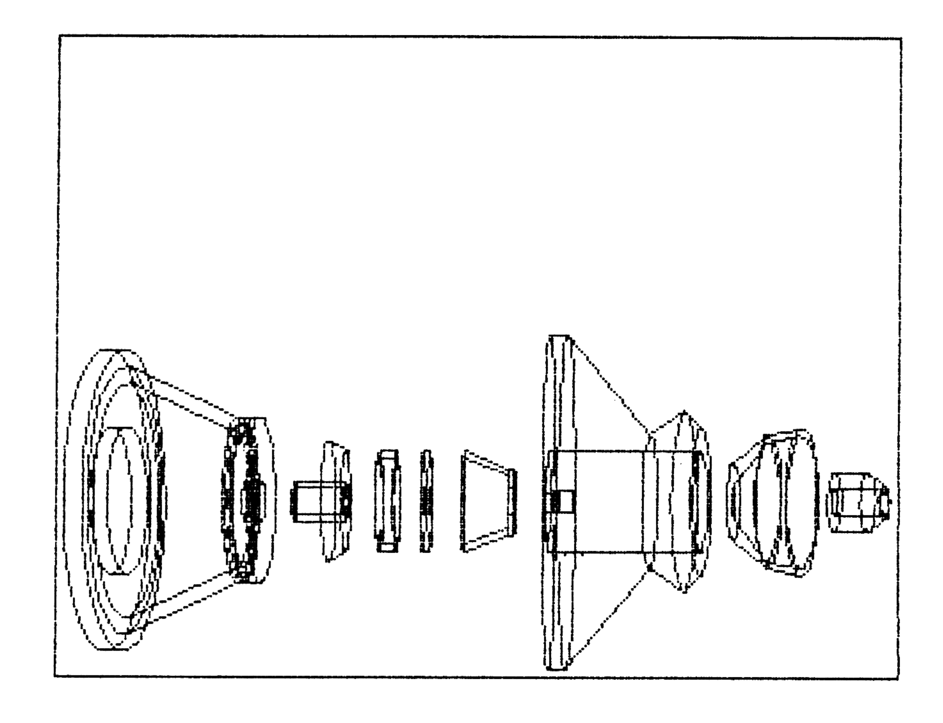


Figure 9.5 — Génération du dessin de la pièce mécanique, en trois dimensions et vue réelle.

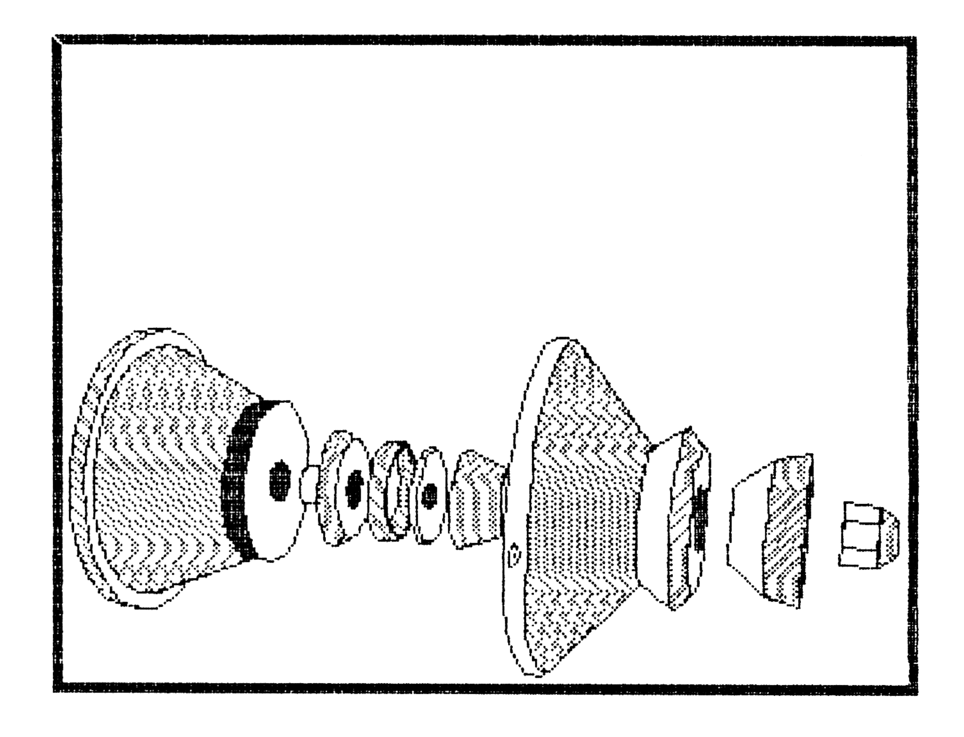


Figure 9.6 — Essai de D.A.O., appliqué au dessin de la pièce mécanique.

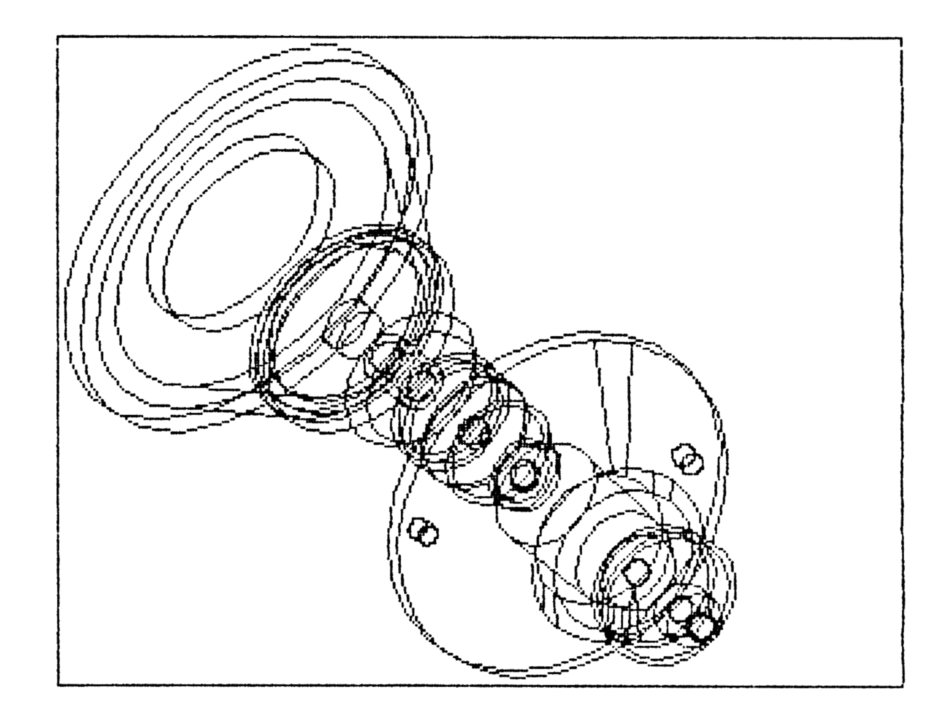


Figure 9.7 — Pièce mécanique représentée en trois dimensions et vue réelle. Pour le relief, image de gauche.

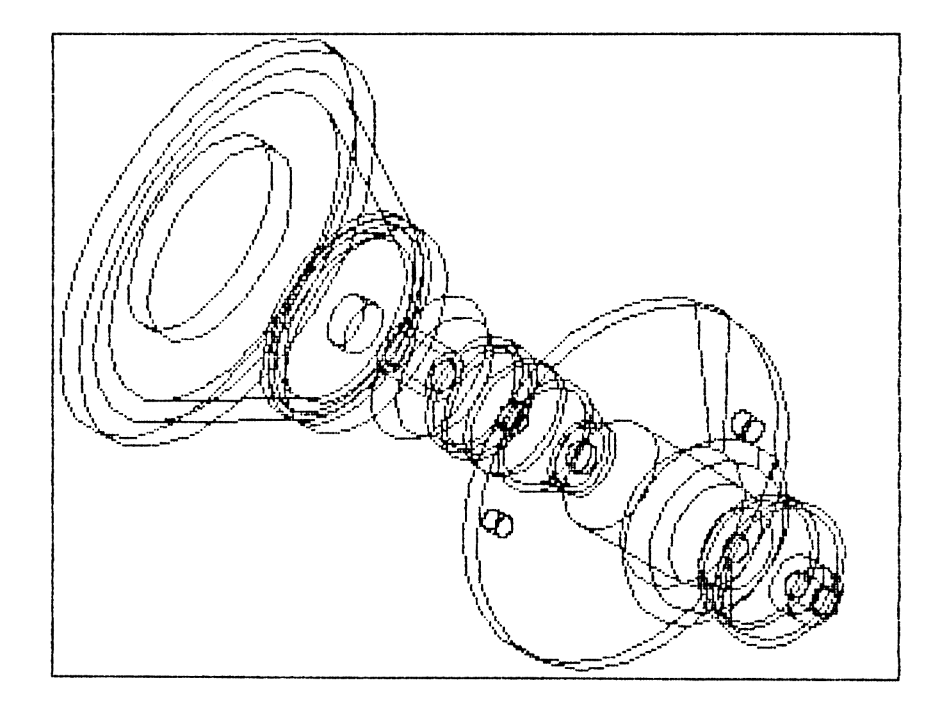


Figure 9.8 — Pièce mécanique représentée en trois dimensions et vue réelle. Pour le relief, image de droite (par modification de l'angle THETA).

Le relief en plus

Dans le cas où l'utilisateur désirerait obtenir davantage d'effet tridimensionnel, il est bien évidemment possible de représenter cette pièce en **relief**. Pour cela deux vues différentes doivent être générées, décalées de quelques degrés, en ce qui concerne l'angle **THETA**. La figure 9.7 a été tracée en fixant comme paramètres :

RHO=400
THETA=−60
PHI=−30

Amélioration

Le second tracé (figure 9.8), est identique, sauf en ce qui concerne l'angle **THETA**, qui a été fixé à **−50** degrés. En appliquant l'une des méthodes de visualisation en relief, la situation spatiale des différentes pièces sera perçue d'une façon saisissante. En effaçant les lignes cachées, le résultat serait naturellement encore amélioré.

Dans le domaine de la mécanique, l'important est de bien montrer toutes les composantes des différentes pièces ; à l'inverse du dessin artistique. Par contre, l'essentiel consiste bien à pouvoir représenter aisément n'importe quelle forme, ou ensemble de formes, quel que soit le point de vision.

3. LES JEUX

En poursuivant les recherches au niveau des applications possibles, dans le domaine des tracés en trois dimensions et en relief, celui dédié aux jeux occupe une place importante. L'époque des jeux simplistes est bien révolue et désormais les utilisateurs sont devenus exigeants, à juste titre. Il faut reconnaître que l'évolution du matériel permet de nos jours la création de superbes décors, en haute résolution et en couleur. Un logiciel de **D.A.O.** peut donc réellement être utile aux concepteurs de programmes ludiques, en leur permettant de ne plus se préocuper des lois de la représentation spatiale des objets.

Afin d'illustrer notre propos, c'est le thème d'un jeu de l'espace qui a été sélectionné. La figure 9.9 en montre le plan d'ensemble, tracé manuellement sur papier millimétré, selon la méthode désormais classique. Puis les différentes coordonnées, composant le décor, seront reportées au sein du programme habituel, comme ci-dessous.

Encore un « Envahisseurs ».

```
2360 '====================  J E U X  ======================
2370 '
2380 'FRONT GAUCHE
2390 DATA  75, -50,  15, 0
2400 DATA  75, -20,  15, 1
2410 DATA  75, -20,  -5, 1
2420 DATA  75,  20,  -5, 1
2430 DATA  75,  20,  15, 1
2440 DATA  75,  50,  15, 1
2450 DATA  75,  50, -15, 1
2460 DATA  75, -50, -15, 1
2470 DATA  75, -50,  15, 1
2480 '
```

Remplace les lignes de « data » du programme précédent.

Listage 9.3 ▶▶▶

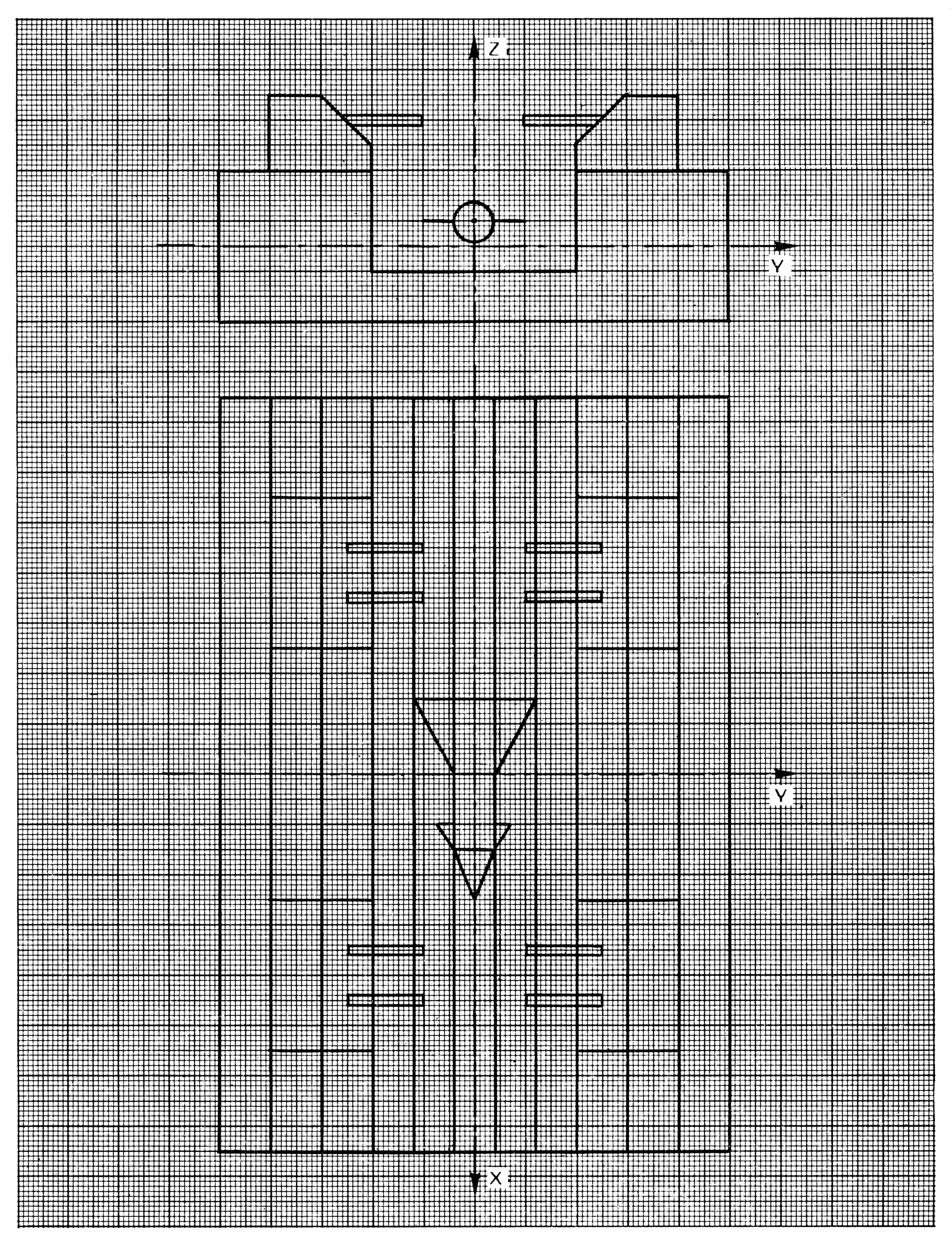


Figure 9.9 — Plan d'une scène de jeu spatial, tracé sur papier millimétré.

▶▶▶

```
2490 'FRONT DROIT
2500 DATA -75, -50,  15, 0
2510 DATA -75, -20,  15, 1
2520 DATA -75, -20,  -5, 1
2530 DATA -75,  20,  -5, 1
2540 DATA -75,  20,  15, 1
2550 DATA -75,  50,  15, 1
2560 DATA -75,  50, -15, 1
2570 DATA -75, -50, -15, 1
2580 DATA -75, -50,  15, 1
2590 '
2600 'LIGNES
2610 DATA  75, -50, -15, 0
2620 DATA -75, -50, -15, 1
2630 DATA  75,  50, -15, 0
2640 DATA -75,  50, -15, 1
2650 '
2660 DATA  75, -50,  15, 0
2670 DATA -75, -50,  15, 1
2680 DATA  75, -20,  15, 0
2690 DATA -75, -20,  15, 1
2700 DATA  75, -20,  -5, 0
2710 DATA -75, -20,  -5, 1
2720 DATA  75,  20,  -5, 0
2730 DATA -75,  20,  -5, 1
2740 DATA  75,  20,  15, 0
2750 DATA -75,  20,  15, 1
2760 DATA  75,  50,  15, 0
2770 DATA -75,  50,  15, 1
2780 '
2790 DATA  75, -40,  15, 0
2800 DATA -75, -40,  15, 1
2810 DATA  75, -30,  15, 0
2820 DATA -75, -30,  15, 1
2830 '
2840 DATA  75,  30,  15, 0
2850 DATA -75,  30,  15, 1
2860 DATA  75,  40,  15, 0
2870 DATA -75,  40,  15, 1
2880 '
2890 DATA  75, -12,  -5, 0
2900 DATA -75, -12,  -5, 1
2910 DATA  75,  -4,  -5, 0
2920 DATA -75,  -4,  -5, 1
2930 DATA  75,   4,  -5, 0
2940 DATA -75,   4,  -5, 1
2950 DATA  75,  12,  -5, 0
2960 DATA -75,  12,  -5, 1
2970 '
```

Beaucoup de lignes avec cette simulation...

```
2980 'BLOC 1
2990 DATA  55, -40,  15, 0
3000 DATA  55, -40,  30, 1
3010 DATA  55, -30,  30, 1
3020 DATA  55, -20,  20, 1
3030 DATA  55, -20,  15, 1
3040 DATA  55, -40,  15, 1
3050 '
3060 DATA  25, -40,  15, 0
3070 DATA  25, -40,  30, 1
3080 DATA  25, -30,  30, 1
3090 DATA  25, -20,  20, 1
3100 DATA  25, -20,  15, 1
3110 DATA  25, -40,  15, 1
3120 '
3130 DATA  55, -40,  30, 0
3140 DATA  25, -40,  30, 1
3150 DATA  55, -30,  30, 0
3160 DATA  25, -30,  30, 1
3170 DATA  55, -20,  20, 0
3180 DATA  25, -20,  20, 1
3190 '
3200 'BLOC 2
3210 DATA -25, -40,  15, 0
3220 DATA -25, -40,  30, 1
3230 DATA -25, -30,  30, 1
3240 DATA -25, -20,  20, 1
3250 DATA -25, -20,  15, 1
3260 DATA -25, -40,  15, 1
3270 '
3280 DATA -55, -40,  15, 0
3290 DATA -55, -40,  30, 1
3300 DATA -55, -30,  30, 1
3310 DATA -55, -20,  20, 1
3320 DATA -55, -20,  15, 1
3330 DATA -55, -40,  15, 1
3340 '
3350 DATA -25, -40,  30, 0
3360 DATA -55, -40,  30, 1
3370 DATA -25, -30,  30, 0
3380 DATA -55, -30,  30, 1
3390 DATA -25, -20,  20, 0
3400 DATA -55, -20,  20, 1
3410 '
```

Listage 9.3 ▶▶▶

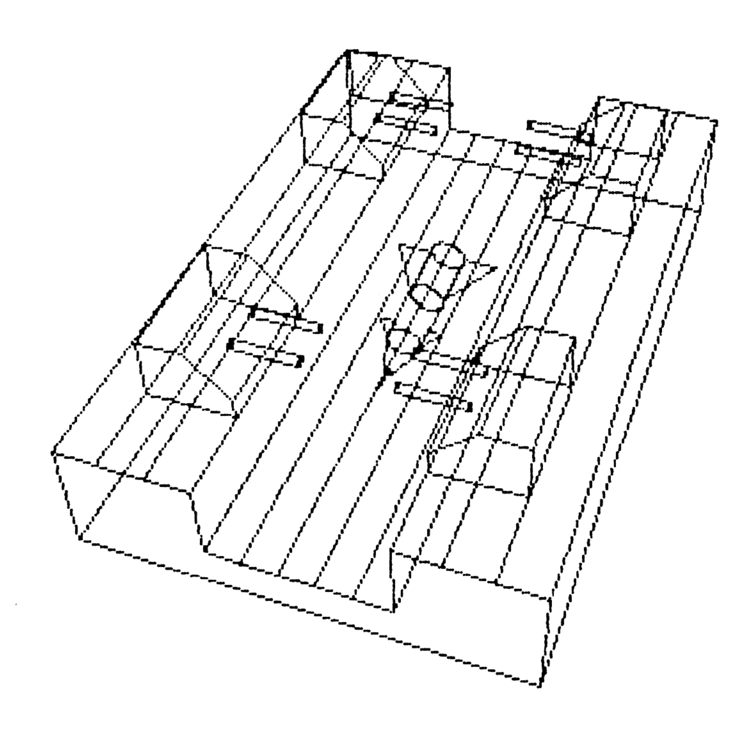


Figure 9.10 — Génération « brute » de la scène du jeu spatial, en trois dimensions et vue réelle.

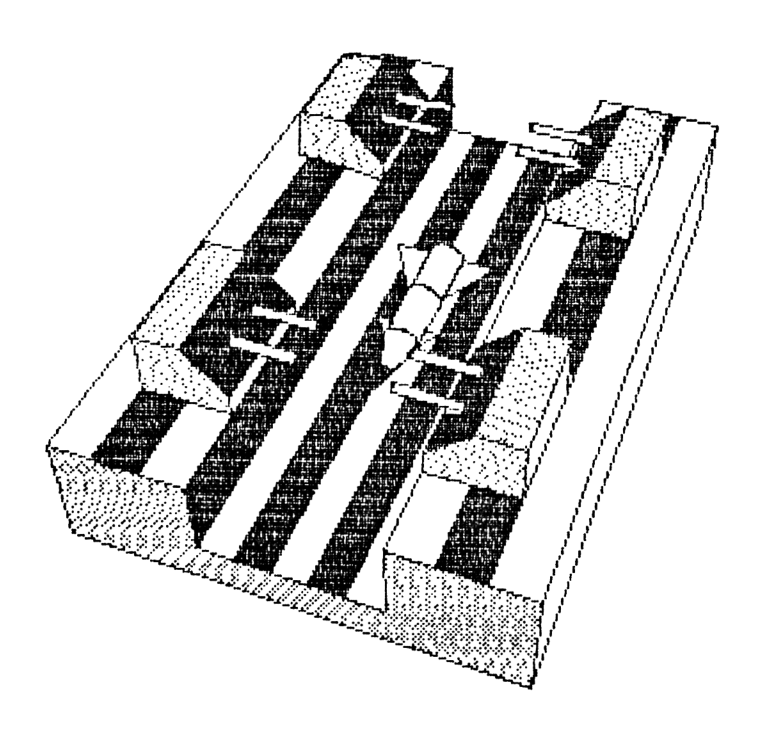


Figure 9.11 — Exemple d'amélioration du dessin, par D.A.O.

```
▶▶▶
3420 'BLOC 3
3430 DATA -25,  40,  15, 0
3440 DATA -25,  40,  30, 1
3450 DATA -25,  30,  30, 1
3460 DATA -25,  20,  20, 1
3470 DATA -25,  20,  15, 1
3480 DATA -25,  40,  15, 1
3490 '
3500 DATA -55,  40,  15, 0
3510 DATA -55,  40,  30, 1
3520 DATA -55,  30,  30, 1
3530 DATA -55,  20,  20, 1
3540 DATA -55,  20,  15, 1
3550 DATA -55,  40,  15, 1
3560 '
3570 DATA -25,  40,  30, 0
3580 DATA -55,  40,  30, 1
3590 DATA -25,  30,  30, 0
3600 DATA -55,  30,  30, 1
3610 DATA -25,  20,  20, 0
3620 DATA -55,  20,  20, 1
3630 '
3640 'BLOC 4
3650 DATA  55,  40,  15, 0
3660 DATA  55,  40,  30, 1
3670 DATA  55,  30,  30, 1
3680 DATA  55,  20,  20, 1
3690 DATA  55,  20,  15, 1
3700 DATA  55,  40,  15, 1
3710 '
3720 DATA  25,  40,  15, 0
3730 DATA  25,  40,  30, 1
3740 DATA  25,  30,  30, 1
3750 DATA  25,  20,  20, 1
3760 DATA  25,  20,  15, 1
3770 DATA  25,  40,  15, 1
3780 '
3790 DATA  55,  40,  30, 0
3800 DATA  25,  40,  30, 1
3810 DATA  55,  30,  30, 0
3820 DATA  25,  30,  30, 1
3830 DATA  55,  20,  20, 0
3840 DATA  25,  20,  20, 1
3850 '
3860 'CANONS
3870 DATA  45, -25,  25, 3, 1, 3
3880 DATA  45, -10,  25, 3, 1, 3
3890 DATA  35, -25,  25, 3, 1, 3
3900 DATA  35, -10,  25, 3, 1, 3
3910 '
```

Mais aussi des cylindres, tracés à l'aide de deux cercles et leurs tangentes.

```
3920 DATA -45, -25,  25, 3, 1, 3
3930 DATA -45, -10,  25, 3, 1, 3
3940 DATA -35, -25,  25, 3, 1, 3
3950 DATA -35, -10,  25, 3, 1, 3
3960 '
3970 DATA -45,  25,  25, 3, 1, 3
3980 DATA -45,  10,  25, 3, 1, 3
3990 DATA -35,  25,  25, 3, 1, 3
4000 DATA -35,  10,  25, 3, 1, 3
4010 '
4020 DATA  45,  25,  25, 3, 1, 3
4030 DATA  45,  10,  25, 3, 1, 3
4040 DATA  35,  25,  25, 3, 1, 3
4050 DATA  35,  10,  25, 3, 1, 3
4060 '
4070 'MISSILE
4080 DATA -15,   0,   5, 3, 4, 2
4090 DATA   0,   0,   5, 3, 4, 2
4100 DATA   0,   0,   5, 3, 4, 2
4110 DATA  15,   0,   5, 3, 4, 2
4120 DATA  15,   0,   5, 3, 4, 2
4130 DATA  25,   0,   5, 3, 1, 2
4140 DATA  25,   0,   5, 0
4150 DATA  30,   0,   5, 1
4160 '
4170 DATA -15, -12,   5, 0
4180 DATA -15,  -4,   5, 1
4190 DATA   0,  -4,   5, 1
4200 DATA -15, -12,   5, 1
4210 '
4220 DATA -15,  12,   5, 0
4230 DATA -15,   4,   5, 1
4240 DATA   0,   4,   5, 1
4250 DATA -15,  12,   5, 1
4260 '
4270 DATA  10,  -7,   5, 0
4280 DATA  10,  -4,   5, 1
4290 DATA  15,  -4,   5, 1
4300 DATA  10,  -7,   5, 1
4310 '
4320 DATA  10,   7,   5, 0
4330 DATA  10,   4,   5, 1
4340 DATA  15,   4,   5, 1
4350 DATA  10,   7,   5, 1
4360 '
4370 'FIN
4380 DATA 0, 0, 0, 4
```

Un mélange de toutes les techniques !

Indique la fin des données. A ne pas oublier.

Fin du Listage 9.3

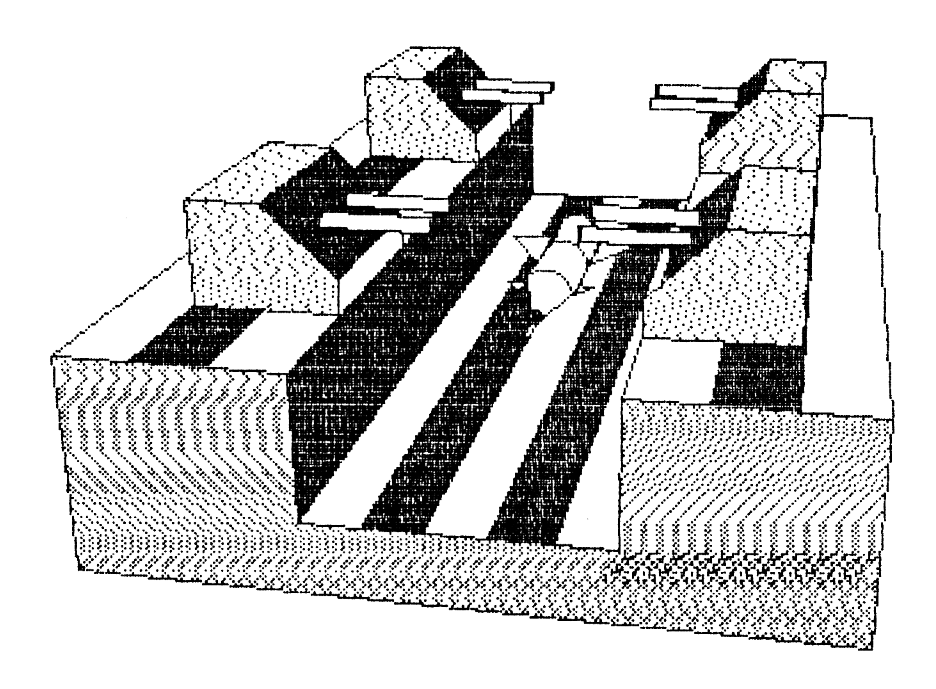


Figure 9.12 — Autre exemple, avec abaissement du point de vue de l'observateur.

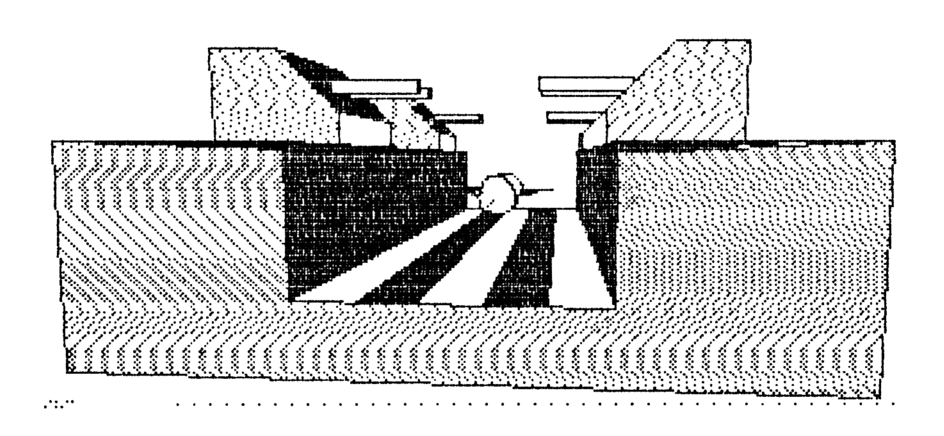


Figure 9.13 — Encore un exemple, le point de vue étant davantage abaissé.

D.A.O. ou J.A.O. ?

Le résultat brut est montré à la figure 9.10, mais il peut être énormément amélioré, en employant un logiciel graphique approprié, comme expliqué au chapitre précédent. Des exemples de réalisations se rapportent aux figures 9.11 à 9.13, bien qu'il soit possible d'atteindre encore plus de réalisme dans la finesse des détails. Néanmoins elles montrent les résultats qu'il est possible d'obtenir, dans ce domaine.

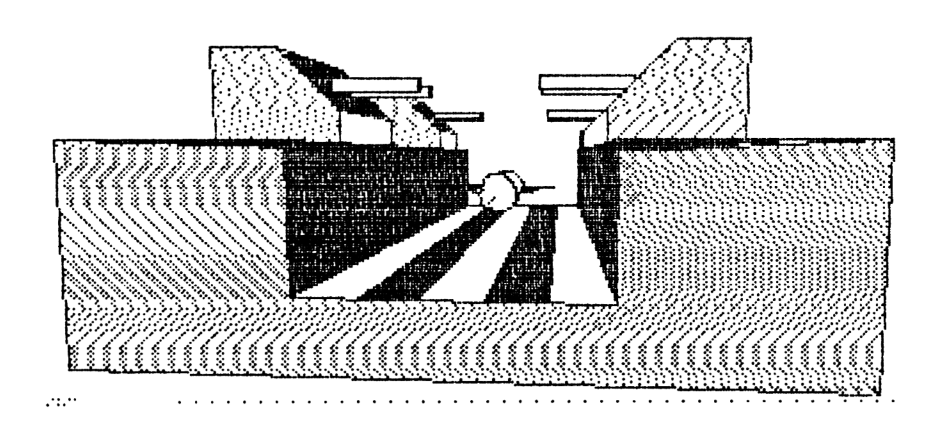


Figure 9.14 — Vue de gauche, destinée à la visualisation en relief, de la scène du jeu spatial.

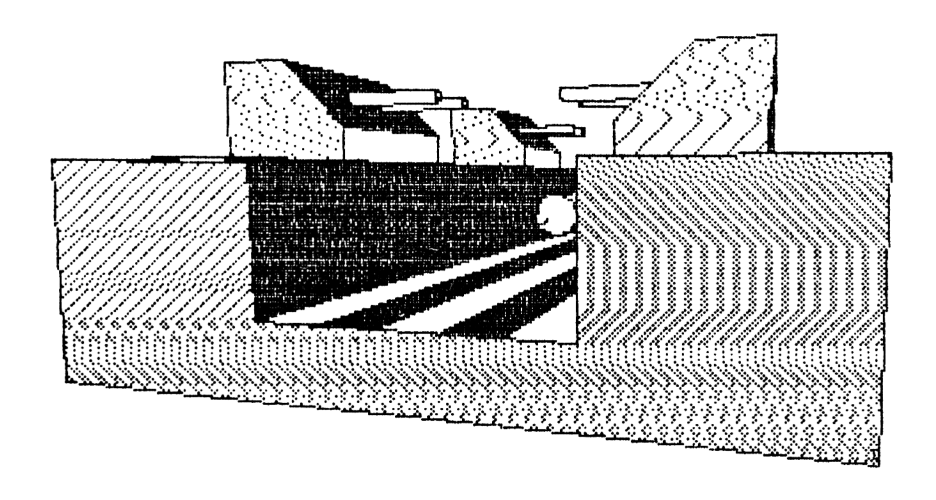


Figure 9.15 — Vue de droite, destinée à la visualisation en relief, de la scène du jeu spatial. L'angle THETA a été augmenté de 10 degrés.

Le thème du jeu pourrait par exemple en être le suivant : vous venez de recevoir pour mission de défendre la galaxie des envahisseurs, qui projettent de détruire la terre. Seul l'anéantissement de leur « Q.G. » pourrait sauver le monde, mais naturellement un bouclier magnétique extrêmement efficace en défend l'accès. Seulement, comme vous êtes courageux, vous n'hésitez pas à vous lancer à l'assaut d'une station orbitale, comportant un couloir d'accès possible, aux commandes de votre vaisseau spatial. Vous remarquerez que cette station est particulièrement bien défendue par quatre *casemates* bien armées, ainsi que par un puissant *missile*. A vous de tenter votre chance, afin de forcer ce passage et sauver toute la galaxie.



Les figures 9.11 à 9.13 indiquent l'action menée: une approche en piqué rapide, en vue d'une attaque au rayon *laser.* Au niveau de la figure 9.11, la base est survolée. Puis le piqué est dirigé vers l'entrée du couloir (figure 9.12). Enfin l'attaque est imminente, avec la figure 9.13 : **feu !**

Par contre, dans ce domaine, l'emploi du **relief** peut apporter des solutions nouvelles au réalisme des simulations. Les figures 9.14 et 9.15 ont été générées dans ce sens. Leur angle **THETA** diffère d'une dizaine de degrés. Elles peuvent être observées selon les méthodes étudiées jusqu'à présent. De cette façon l'effet de *réalisme* est saisissant, l'utilisateur ayant réellement l'impression d'être au coeur de l'action.

Mais le domaine d'application peut varier énormément, dans ce domaine. Il peut concerner réellement n'importe quel ensemble de formes, qu'il s'agisse de tout décor ludique ou même de paysages. Les seules limites sont encore celles de l'imagination et il ne faut pas hésiter, de même, à expérimenter largement.

4. LES HISTOGRAMMES 3D

Voici enfin le terme de nos investigations au niveau de la recherche d'applications en trois dimensions et en relief. Ce qui ne veut pas dire que nous aurons été exhaustifs, bien au contraire. Le dernier domaine que nous allons aborder est celui de la **gestion**, avec les *histogrammes* en trois dimensions.

Graphiques en trois dimensions

C'est peut être ce type d'application qui a propulsé, à une certaine époque, un logiciel intégré comme « Open Access » vers les cimes. Or, en utilisant notre programme habituel (8.1), il est possible de réaliser exactement la même fonction. La figure 9.16 en montre le plan, tracé sur du papier millimétré. Nous désirons générer un graphique en trois dimensions, montrant des successions de *bâtons* représentant des valeurs de données, ainsi que leurs échelles graphiques. Les coordonnées correspondantes sont reportées dans les lignes de *DATA* suivantes, selon la méthode habituelle.

```
2360 '==============  H I S T O G R A M M E S  ==============
2370 '
2380 'FOND
2390 DATA -50, -70, -40, 0
2400 DATA -50,  70, -40, 1
2410 DATA -30, -70, -40, 0
2420 DATA -30,  70, -40, 1
2430 DATA -10, -70, -40, 0
2440 DATA -10,  70, -40, 1
2450 DATA  10, -70, -40, 0
2460 DATA  10,  70, -40, 1
2470 DATA  30, -70, -40, 0
2480 DATA  30,  70, -40, 1
2490 DATA  50, -70, -40, 0
2500 DATA  50,  70, -40, 1
2510 '
```

Ce sont les nouvelles « data » du programme précédent.

Listage 9.4 ▶▶▶

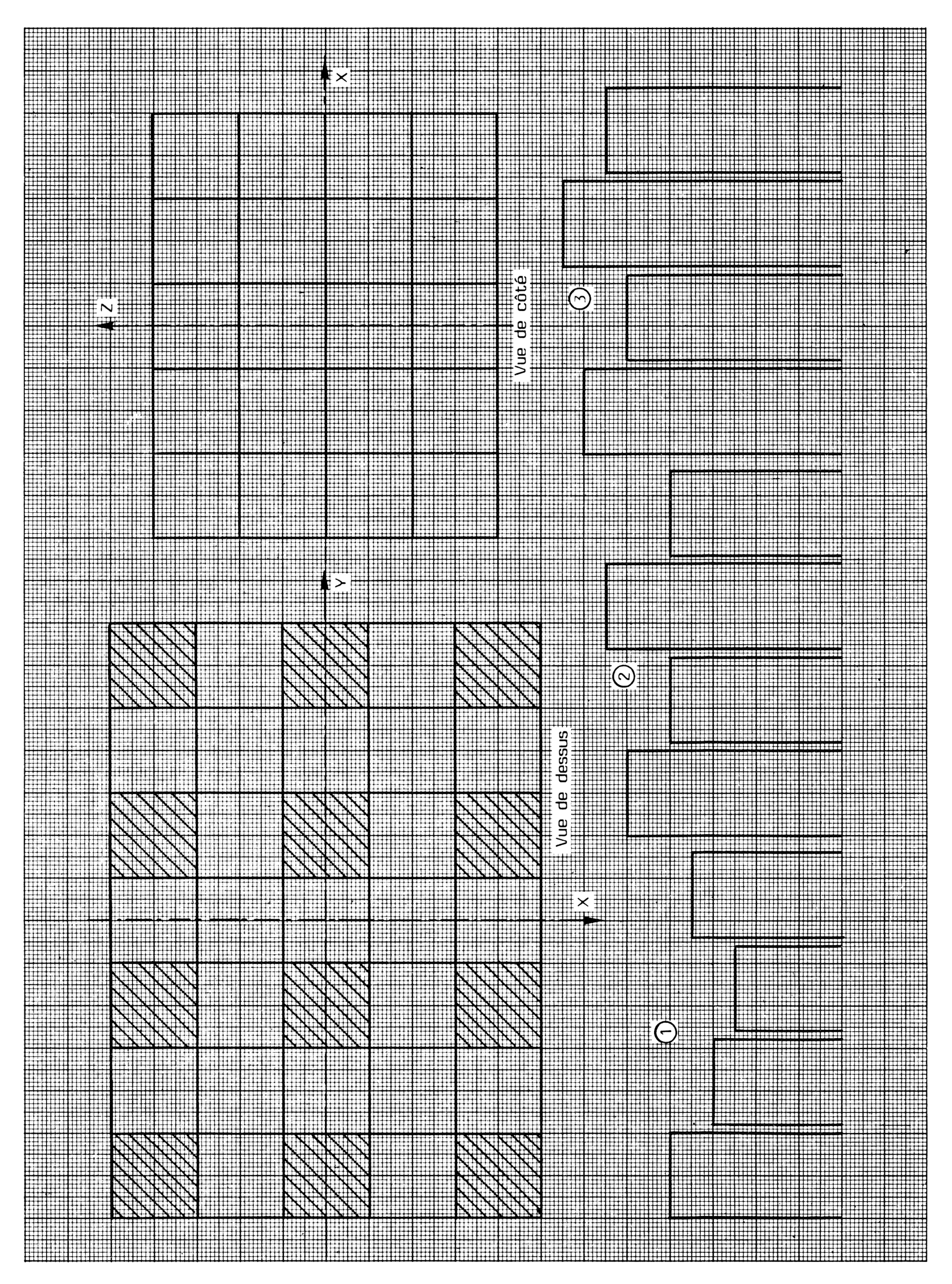


Figure 9.16 — Plan d'un graphique de type bâtons, tracé sur papier millimétré.

```
▶▶▶
2520 DATA  50, -70, -40, 0
2530 DATA -50, -70, -40, 1
2540 DATA  50, -50, -40, 0
2550 DATA -50, -50, -40, 1
2560 DATA  50, -30, -40, 0
2570 DATA -50, -30, -40, 1
2580 DATA  50, -10, -40, 0
2590 DATA -50, -10, -40, 1
2600 DATA  50,  10, -40, 0
2602 DATA -50,  10, -40, 1
2604 DATA  50,  30, -40, 0
2606 DATA -50,  30, -40, 1
2608 DATA  50,  50, -40, 0
2610 DATA -50,  50, -40, 1
2612 DATA  50,  70, -40, 0
2614 DATA -50,  70, -40, 1
2616 '
2618 'COTE GAUCHE
2620 DATA -50, -70, -20, 0
2622 DATA  50, -70, -20, 1
2624 DATA -50, -70,   0, 0
2626 DATA  50, -70,   0, 1
2628 DATA -50, -70,  20, 0
2630 DATA  50, -70,  20, 1
2632 DATA -50, -70,  40, 0
2634 DATA  50, -70,  40, 1
2636 '
2638 DATA -50, -70, -40, 0
2640 DATA -50, -70,  40, 1
2642 DATA -30, -70, -40, 0
2644 DATA -30, -70,  40, 1
2646 DATA -10, -70, -40, 0
2648 DATA -10, -70,  40, 1
2650 DATA  10, -70, -40, 0
2652 DATA  10, -70,  40, 1
2654 DATA  30, -70, -40, 0
2656 DATA  30, -70,  40, 1
2658 DATA  50, -70, -40, 0
2660 DATA  50, -70,  40, 1
2662 '
2664 'ARRIERE
2666 DATA -50, -70, -20, 0
2668 DATA -50,  70, -20, 1
2670 DATA -50, -70,   0, 0
2672 DATA -50,  70,   0, 1
2674 DATA -50, -70,  20, 0
2676 DATA -50,  70,  20, 1
2678 DATA -50, -70,  40, 0
2680 DATA -50,  70,  40, 1
2682 '
```

On trace d'abord le cadre tridimensionnel.

```
2684 DATA -50, -50, -40, 0
2686 DATA -50, -50,  40, 1
2688 DATA -50, -30, -40, 0
2690 DATA -50, -30,  40, 1
2692 DATA -50, -10, -40, 0
2694 DATA -50, -10,  40, 1
2696 DATA -50,  10, -40, 0
2698 DATA -50,  10,  40, 1
2700 DATA -50,  30, -40, 0
2702 DATA -50,  30,  40, 1
2704 DATA -50,  50, -40, 0
2706 DATA -50,  50,  40, 1
2708 DATA -50,  70, -40, 0
2710 DATA -50,  70,  40, 1
2712 '
2714 'RANGEE ARRIERE
2716 DATA -50, -70,  20, 0
2718 DATA -50, -50,  20, 1
2720 DATA -30, -50,  20, 1
2722 DATA -30, -70,  20, 1
2724 DATA -50, -70,  20, 1
2726 DATA -30, -50,  20, 0
2728 DATA -30, -50, -40, 1
2730 '
2732 DATA -50, -30,  10, 0
2734 DATA -50, -10,  10, 1
2736 DATA -30, -10,  10, 1
2738 DATA -30, -30,  10, 1
2740 DATA -50, -30,  10, 1
2742 DATA -30, -10,  10, 0
2744 DATA -30, -10, -40, 1
2746 DATA -30, -30,  10, 0
2748 DATA -30, -30, -40, 1
2750 '
2752 DATA -50,  10,  25, 0
2754 DATA -50,  30,  25, 1
2756 DATA -30,  30,  25, 1
2758 DATA -30,  10,  25, 1
2760 DATA -50,  10,  25, 1
2762 DATA -30,  10,  25, 0
2764 DATA -30,  10, -40, 1
2766 DATA -30,  30,  25, 0
2768 DATA -30,  30, -40, 1
2770 '
```

Puis les « cubes » représentant les données.

Listage 9.4 ▶▶▶

▶▶▶

```
3000 DATA -50,  50,  15, 0
3150 DATA -50,  70,  15, 1
3160 DATA -30,  70,  15, 1
3170 DATA -30,  50,  15, 1
3175 DATA -50,  50,  15, 1
3180 DATA -30,  50,  15, 0
3190 DATA -30,  50, -40, 1
3200 DATA -30,  70,  15, 0
3210 DATA -30,  70, -40, 1
3220 '
3230 'RANGEE DU MILIEU
3240 DATA -10, -70,  10, 0
3250 DATA -10, -50,  10, 1
3260 DATA  10, -50,  10, 1
3270 DATA  10, -70,  10, 1
3275 DATA -10, -70,  10, 1
3280 DATA -10, -50,  10, 0
3290 DATA -10, -50, -40, 1
3300 DATA  10, -50,  10, 0
3310 DATA  10, -50, -40, 1
3320 '
3330 DATA -10, -30,   5, 0
3340 DATA -10, -10,   5, 1
3350 DATA  10, -10,   5, 1
3360 DATA  10, -30,   5, 1
3370 DATA -10, -30,   5, 1
3380 DATA -10, -30,   5, 0
3390 DATA -10, -30, -40, 1
3400 DATA -10, -10,   5, 0
3410 DATA -10, -10, -40, 1
3420 DATA  10, -10,   5, 0
3430 DATA  10, -10, -40, 1
3440 DATA  10, -30,   5, 0
3450 DATA  10, -30, -40, 1
3460 '
3470 DATA -10,  10,  10, 0
3480 DATA -10,  30,  10, 1
3490 DATA  10,  30,  10, 1
3500 DATA  10,  10,  10, 1
3510 DATA -10,  10,  10, 1
3520 DATA -10,  10,  10, 0
3530 DATA -10,  10, -40, 1
3540 DATA -10,  30,  10, 0
3550 DATA -10,  30, -40, 1
3560 DATA  10,  30,  10, 0
3570 DATA  10,  30, -40, 1
3580 DATA  10,  10,  10, 0
3590 DATA  10,  10, -40, 1
```

On passe à une autre rangée de « cubes ».

```
3600 '
3610 DATA -10,  50,   0, 0
3620 DATA -10,  70,   0, 1
3630 DATA  10,  70,   0, 1
3640 DATA  10,  50,   0, 1
3650 DATA -10,  50,   0, 1
3660 DATA -10,  50,   0, 0
3670 DATA -10,  50, -40, 1
3680 DATA -10,  70,   0, 0
3690 DATA -10,  70, -40, 1
3700 DATA  10,  70,   0, 0
3710 DATA  10,  70, -40, 1
3720 DATA  10,  50,   0, 0
3730 DATA  10,  50, -40, 1
3740 '
3750 'RANGEE AVANT
3760 DATA  30, -70,   0, 0
3770 DATA  30, -50,   0, 1
3780 DATA  50, -50,   0, 1
3790 DATA  50, -70,   0, 1
3795 DATA  30, -70,   0, 1
3800 DATA  30, -50,   0, 0
3810 DATA  30, -50, -40, 1
3820 DATA  50, -50,   0, 0
3830 DATA  50, -50, -40, 1
3840 '
3850 DATA  30, -30, -10, 0
3860 DATA  30, -10, -10, 1
3870 DATA  50, -10, -10, 1
3880 DATA  50, -30, -10, 1
3890 DATA  30, -30, -10, 1
3900 DATA  30, -30, -10, 0
3910 DATA  30, -30, -40, 1
3920 DATA  30, -10, -10, 0
3930 DATA  30, -10, -40, 1
3940 DATA  50, -10, -10, 0
3950 DATA  50, -10, -40, 1
3960 DATA  50, -30, -10, 0
3970 DATA  50, -30, -40, 1
```

Pour terminer par la dernière... qui est la première !

Listage 9.4 ▶▶▶

▶▶▶

```
3980 '
3990 DATA  30,  10,   -5, 0
4000 DATA  30,  30,   -5, 1
4010 DATA  50,  30,   -5, 1
4020 DATA  50,  10,   -5, 1
4030 DATA  30,  10,   -5, 1
4040 DATA  30,  10,   -5, 0
4050 DATA  30,  10,  -40, 1
4060 DATA  30,  30,   -5, 0
4070 DATA  30,  30,  -40, 1
4080 DATA  50,  30,   -5, 0
4090 DATA  50,  30,  -40, 1
4100 DATA  50,  10,   -5, 0
4110 DATA  50,  10,  -40, 1
```

```
4120 '
4130 DATA  30,   50, -10, 0
4140 DATA  30,   70, -10, 1
4150 DATA  50,   70, -10, 1
4160 DATA  50,   50, -10, 1
4170 DATA  30,   50, -10, 1
4180 DATA  30,   50, -10, 0
4190 DATA  30,   50, -40, 1
4200 DATA  30,   70, -10, 0
4210 DATA  30,   70, -40, 1
4220 DATA  50,   70, -10, 0
4230 DATA  50,   70, -40, 1
4240 DATA  50,   50, -10, 0
4250 DATA  50,   50, -40, 1
4260 '
4270 'FIN
4280 DATA 0, 0, 0, 4
```

Et enfin on n'oublie pas d'indiquer la fin des données...

Fin du listage 9.4

Pour un véritable logiciel de gestion.

Bien entendu, un programme réalisant ce type d'application devrait être modifié. Il faudrait permettre la saisie manuelle des séries de données (figées ici en lignes de *DATA*), voire même leur chargement automatique à partir d'un fichier issu d'un autre logiciel. L'ajustement automatique des dimensions du cadre général devrait être aussi réalisé, en fonction de la valeur maximum atteinte par les données. Enfin, les différentes couleurs des faces devraient pouvoir être définies par l'utilisateur, d'une manière interactive. Malgré l'absence de ces options, la figure 9.17 montre quand même ce qu'il est possible d'obtenir dans ce domaine.

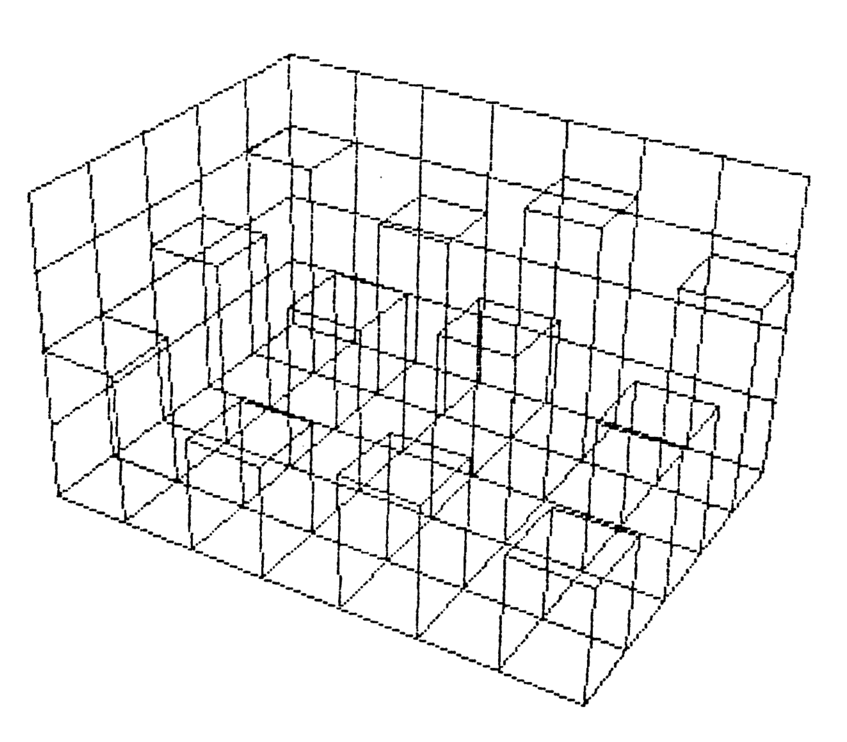


Figure 9.17 — Génération « brute » du graphique de type bâtons, en trois dimensions et vue réelle.

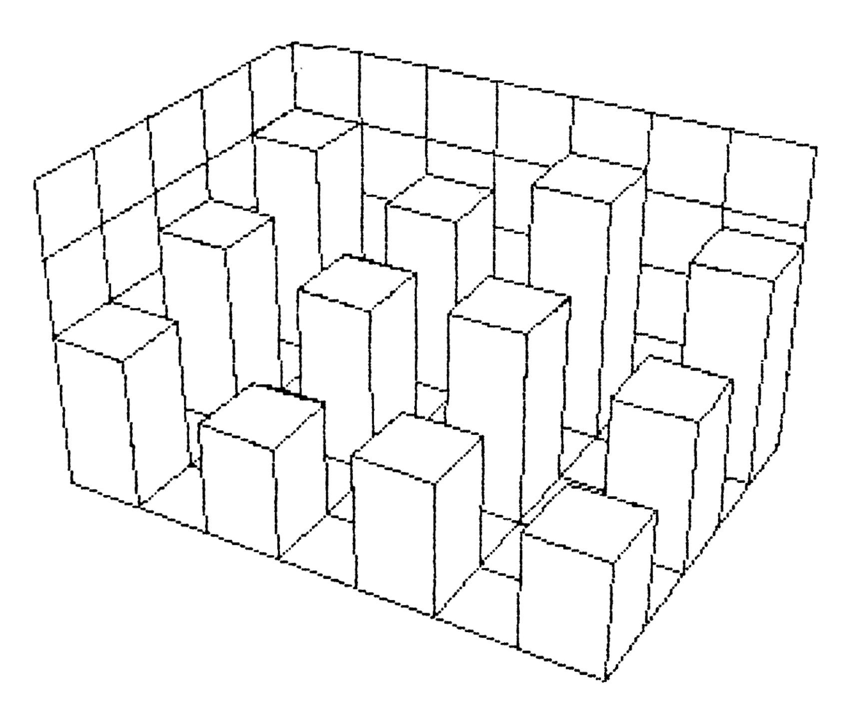


Figure 9.18 — Amélioration de la lisibilité, par effacement des lignes cachées, sur la vue réelle générée par le programme.

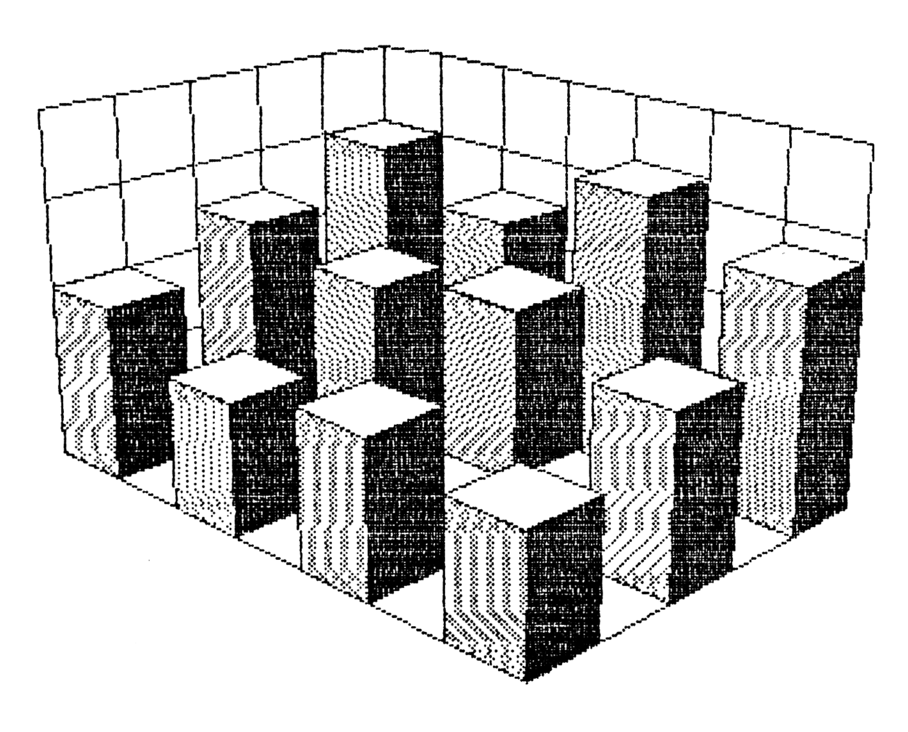


Figure 9.19 — Dernière amélioration de la vue réelle générée par le programme. Pour la visualisation en relief, vue de gauche.



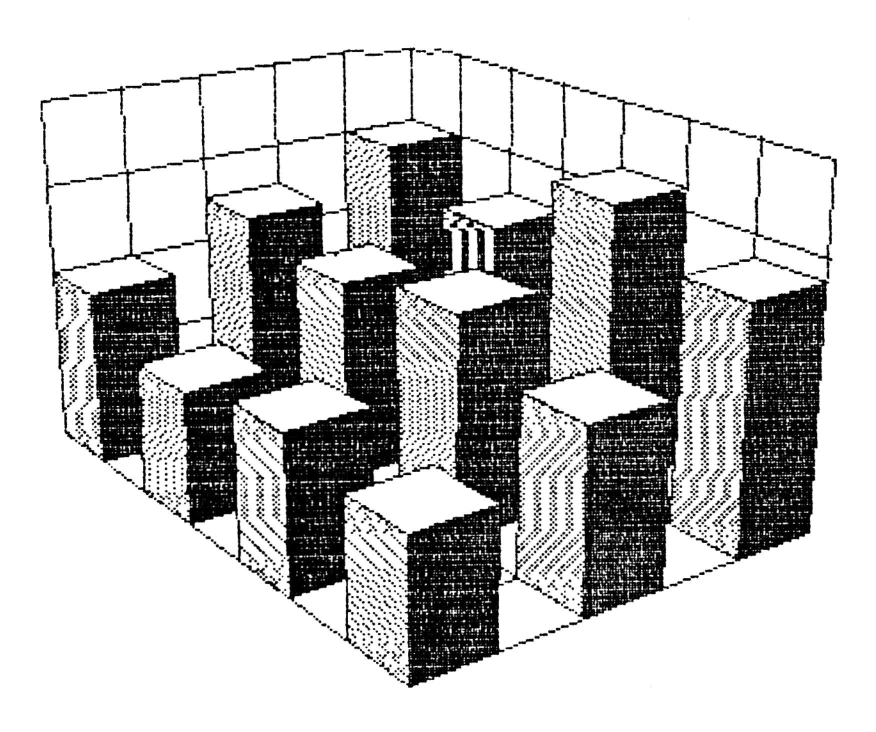


Figure 9.20 — Modifications identiques, mais vue de droite pour le relief, par augmentation de l'angle THETA de 10 degrés.

L'essentiel du travail est effectué par le programme, c'est-à-dire dessiner automatiquement les divers tracés en respectant les lois de la représentation spatiale des objets. Maintenant, il est facile de retoucher le dessin à l'aide d'un logiciel graphique, par exemple. En effaçant simplement les lignes cachées, c'est la figure 9.18 qui a été obtenue.

La figure 9.19 a encore été agrémentée en colorant les faces des bâtons. Elle montre l'ensemble du dessin sous un angle différent, après redéfinission des angles **THETA** et **PHI**. Ce qui nous amène tout naturellement à aborder, une fois de plus, le domaine du *relief*, simplement en générant un deuxième tracé différent du premier. La figure 9.20 a été obtenue en augmentant simplement l'angle **THETA** de **10** degrés, par rapport à celui fixé lors de la figure 9.19 Les méthodes habituelles de visualisation seront employées avec profit, afin d'observer cet histogramme en **relief**, c'est-à-dire en « voyant », avec plus de réalisme, l'état des données entre elles.

Bien d'autres domaines restent encore à explorer, ainsi que toutes sortes de possibilités. Il appartient désormais au lecteur de prendre le relais et d'apporter sa part de réalisations à la communauté informatique. Gageons qu'il y réussira brillamment.

CHAPITRE 10

UTILISATION D'UN LANGAGE EVOLUE

Tout récemment, les données ont été sensiblement modifiées, dans le domaine de la micro-informatique. Plus que l'apparition de matériels de plus en plus performants, c'est surtout le développement de certains *standards*, qui a marqué et qui marquera de plus en plus les niveaux « hard » et « soft » de l'informatique mondiale, avec toutefois davantage d'accuité aux Etats Unis.

IBM-PC

En ce qui concerne le matériel, point n'est besoin de rappeler qu'insidieusement l'**IBM-PC** a institué un véritable standard, dans l'environnement professionnel.

TURBO PASCAL

Tandis qu'au niveau du logiciel, tout récemment une nouvelle implémentation d'un langage universellement reconnu, a vu le jour: le **TURBO PASCAL**. Ce langage est en passe de devenir **le** standard en matière de logiciel de programmation. Aux U.S.A., cette standardisation est déjà effectuée et en France la voie a été tracée, avec le « Plan Informatique pour Tous ».

Ce chapitre sera donc consacré à l'adaptation de certains programmes, aux nouveaux standards, c'est-à-dire à une programmation en **Turbo Pascal** sur **IBM-PC** ou compatibles. Un programme complet en sera développé, à titre d'exemple.

1. LE TURBO PASCAL SUR IBM-PC

Le matériel choisi étant le micro-ordinateur **IBM-PC**, il n'est pas inutile d'en rappeler les principales caractéristiques. Il s'agit d'un appareil à caractère résolument professionnel, mais pouvant néanmoins convenir à des amateurs passionnés de micro-informatique, désirant « aller plus loin », après une formation exhaustive sur des « familiaux ». Il est vrai que dans ce cas ce sont plutôt des *compatibles* qui sont choisis, pour des raisons de prix de revient.

Le micro-processeur utilisé est issu d'**INTEL**. Il s'agit d'un **8088** (*pseudo 16 bits*) pour le **PC**, ou d'un **8086** pour certains compatibles (*vrai 16 bits*). L'horloge de son *unité centrale* « tourne » à **4,77** Mhz pour l'**IBM-PC** et souvent à **8** Mhz, pour les compatibles.

Carte E.G.A.

Au niveau de la haute résolution graphique, diverses options sont possibles. En général les compatibles en offrent une d'origine, adaptée à des moniteurs *couleur* ou *monochrome*. Pour l'**IBM-PC**, le problème est un peu plus complexe. I.B.M. offre une *carte graphique couleur*, qui ne peut être exploitée que par des moniteurs couleur (mode **R.G.B**). Par contre le marché des *O.E.M.* a développé un certain nombre de cartes graphiques, aux caractéristiques intéressantes

Deux cartes graphiques, principalement, jouissent d'une notoriété satisfaisante. La carte **Hercules** *monochrome*, permettant l'utilisation du moniteur monochrome d'origine de l'**IBM-PC** et la carte **M.G.C.**, de **Paradise**. Si la carte Hercules offre une résolution graphique supérieure à celle du standard d'I.B.M., elle n'est cependant pas compatible avec les routines graphiques de **Turbo Pascal**, qui lui ne s'écarte pas du standard.

Par contre la carte **Paradise** offre les choix suivants, particulièrement intéressants: utiliser un *moniteur couleur* ou l'*écran monochrome* standard IBM, en émulation couleur, donc avec **16** tonalités de « gris ». L'utilisation d'un écran couleur ne représentant en moyenne que **30 %** des utilisateurs, cette dernière solution paraît réellement représenter le choix idéal. Ainsi les deux modes sont disponibles, par l'intermédiaire d'un « switch » et la compatibilité avec **Turbo Pascal** est totale. En mode graphique, deux choix seront alors possible: une moyenne résolution de **320** pixels sur **200** en **16** couleurs ou une haute résolution de **640** sur **200** pixels en deux couleurs, le noir étant fixé d'office pour le fond.

Le **Turbo Pascal** est un langage intéressant, en ce qui nous concerne, à plus d'un titre. Sa mise en œuvre est des plus simples, comparable à celle du *BASIC*. Le temps mis pour compiler un programme est rellement le plus rapide qui soit, jusqu'à cent fois plus court qu'avec certains autres compilateurs en Pascal. Dès la première erreur, la compilation s'arrête, un message d'erreur explicite est généré, l'éditeur pleine page est automatiquement activé et le curseur positionné à l'endroit de l'erreur.

Un fichier peut aisément être transformé en format « **.COM** », immédiatement exécutable sous **DOS** par tous, sans possession obligatoire du langage **Turbo Pascal** et sans *royalties*. De plus le code généré est le plus rapide rencontré à ce jour, car il est transformé directement en code *natif*, sans passer par un code intermédiaire.

Les accès disque sont également optimisés et ce langage comporte en standard tout un ensemble de routines graphiques permettant d'exploiter au mieux les caractéristiques de la haute résolution couleur de l'**IBM-PC**. Enfin, il s'agit du compilateur le moins cher du marché, en passe de devenir un véritable standard au niveau mondial.

Le langage idéal

Toutes ces caractéristiques en font le langage idéal, pour l'utilisation de nos programmes graphiques complexes. Ainsi les résultats seront vraiment satisfaisants, aussi bien au niveau de la rapidité d'exécution que de la qualité des représentations graphiques générées. Entrer dans le détail du langage Pascal en général sortirait du cadre de notre étude, par contre il est particulièrement intéressant d'étudier les différentes *primitives* spécifiques à **Turbo Pascal** et principalement celles relevant du domaine graphique.

Le langage offre trois modes d'exploitation de l'écran, choisis en fonction des résultats escomptés: TextMode, GraphColorMode et Hires. Chacun possède ses caractéristiques propres, ainsi qu'un jeu de fonctions approprié.

3 modes d'affichage

TextMode:
Positionne le fonctionnement en mode *texte* uniquement.

TextColor (couleur):
Choisit une couleur parmi *16*, pour les textes.

ClrScr:
Efface l'écran. Le fait de changer de mode de visualisation produit aussi le même effet.

Window (X1, Y1, X2, Y2):
Définit une fenêtre à part entière, dans l'écran, que les textes ne pourront en aucun cas franchir. En cas de dépassement du nombre de lignes alloué, un *scrolling* ne sera effectué qu'au sein de cette fenêtre, laissant le reste de l'écran inchangé.

HiRes:
Force le fonctionnement en mode *haute résolution* avec un fond noir. La résolution atteindra **640×200** *pixels*.

HiResColor (couleur):
Définit la couleur du *tracé*, en mode « HiRes », parmi 16.

GraphColorMode

GraphColorMode:
Ne permet de disposer que de **320×200** pixels, mais avec un choix de plusieurs couleurs, définies par diverses *palettes.* De plus, dans ce mode, les cercles générés par certaines primitives seront parfaits. Pour ces raisons nous avons choisi ce dernier mode, pour notre exemple de réalisation. D'autres primitives graphiques nous seront de même très utiles.

GraphBackGround (couleur):
Fixe la couleur du fond de l'écran, si le mode sélectionné est différent de « HiRes ».

TextColor (couleur):
Définit la couleur des caractères, qui seront *mixables* avec les tracés en haute résolution.

Palette (numéro):
Effectue un choix parmi les quatre palettes de couleurs disponibles.

Color:
Indique la couleur du tracé, au sein d'une *palette.*

ColorTable (0, 1, 2, 3):
Permet de modifier l'ordre des couleurs d'une *palette.*

GraphWindow (X1, Y1, X2, Y2):
Fixe les limites d'une fenêtre dans l'écran, que les graphiques ne pourront jamais franchir. En cas de dépassement, le reste du tracé sera simplement ignoré.

FillScreen (couleur):
Remplit le fond de la fenêtre graphique, avec l'une des couleurs disponibles.

Ces commandes permettent de définir l'environnement de travail. Il reste encore quelques primitives, à utiliser dans notre programme :

GotoXY (x, y):
Amène le curseur à des coordonnées spécifiques, dans la fenêtre (sachant que l'écran peut-être considéré comme une fenêtre, par défaut).

Draw

Draw (X1, Y1, X2, Y2, couleur):
Ce sera la primitive que nous utiliserons principalement pour les tracés proprement dits. Elle permet de tracer un segment de droite, reliant deux points de coordonnées fixées.

Sound (nombre)
Delay (nombre)
NoSound:
Permettent d'introduire un peu de musique, en indiquant la fin de calculs, par exemple. Les autres primitives graphiques ne seront pas utilisées, pour notre exemple.

Il y a encore deux différences fondamentales avec le *BASIC:* la commande **Draw** ne permet pas de spécifier le dernier point tracé en tant que premières coordonnées du tracé suivant. Il faudra donc mettre en réserve les caractéristiques du dernier point tracé. Mais surtout les fichiers de *DATA*, qui ont fait la « célébrité » du *BASIC*, ne peuvent pas être inclus au sein des programmes en **Turbo Pascal**. Il va donc falloir trouver une solution satisfaisante, tant au niveau de la facilité d'utilisation que des performances.

Mieux que les DATAs.

La solution choisie, si elle complique un peu la programmation, a cependant pour résultat de conférer à l'entrée des données une souplesse importante. Les nombres composant les données définies par l'utilisateur seront simplement entrés dans un fichier texte tout à fait classique, sans aucune contrainte d'utilisation.

; indique la fin d'une ligne

En effet, il suffit d'entrer simplement sous l'éditeur de **Turbo Pascal**, comme pour écrire un programme. A ce moment, les nombres seront entrés à la suite les uns des autres, par ligne logique, séparés par un ou plusieurs espaces. Les nombres *réels* auront la virgule matérialisée par un point. Du texte explicatif peut être introduit n'importe où au sein du fichier, sans problèmes et sans limitations particulières. Il faut cependant veiller à terminer chaque ligne de données par un « ; », car l'éditeur n'introduit pas automatiquement une marque de fin de ligne classique, lors du changement de ligne. Il a donc fallu définir un signe permettant de simuler cette fonction. Or, en Pascal, le « ; » est tout naturel pour marquer la fin d'une instruction.

Par contre, une ligne peut commencer par un nombre quelconque d'espaces et deux nombres peuvent être séparés de même par un espace de longueur variable. Cependant au moins **un** blanc doit les séparer. Cette solution permet de cadrer les nombres pour en améliorer la présentation, offre la possibilité de sauter des lignes ou d'introduire des commentaires et cela avec une grande souplesse.

Fonction de lecture syntaxique

Au niveau de la programmation, il faudra réaliser une fonction de lecture syntaxique du fichier, comportant un *mini-interpréteur.* Ce n'est pas difficile et de plus elle pourra resservir au niveau d'autres applications, qui auraient besoin d'exploiter un fichier de *DATA*. Le résultat obtenu est même d'un emploi plus souple que celui des lignes de *DATA* du *BASIC* et réellement très simple à mettre en œuvre.

2. EXEMPLE D'ADAPTATION

C'est le programme relatif à la **C.A.O.** qui a été choisi pour être adapté à l'**IBM-PC**, à l'aide du **Turbo Pascal**. Le *listage* du programme « CAO », ci-dessous, montre que d'autres fonctions ont été ajoutées et certaines parties du programme modifiées. Ceci non seulement afin de tenir compte du contexte lié à ce matériel et à ce langage, mais aussi dans le but d'apporter certaines améliorations non négligeables.

A la suite de ce listage, trois exemples de fichiers de données sont imprimés: tracés d'*avions*, de *dessins géométriques* et de *pièces mécaniques.* La certaine spécificité, qui a été introduite avec ces réalisations, est néanmoins acceptable du fait que l'**IBM-PC**, ainsi que le **Turbo Pascal**, sont devenus de véritables *standards.*

La principale modification se situe au niveau du dialogue avec l'utilisateur, qui a été rendu beaucoup plus *convivial.* Puis, étant donné que le fait de sélectionner le mode haute résolution efface l'écran, il a fallu mettre au point une boucle d'entrée permettant de stocker l'ensemble du travail à effectuer ultérieurement. Enfin, une analyse syntaxique classique de fichier texte, jointe à un *mini-interpréteur*, a énormément facilité l'entrée des données.

Le programme commence par lancer la procédure « Check », qui a pour but d'éviter l'utilisation du logiciel sur une machine dépourvue de carte graphique. Puis la procédure « Entrees » enchaîne les différents écrans de saisie. Tout d'abord au niveau global, en demandant le nom du fichier à exécuter. Dans le cas où celui-ci n'existerait pas, un message d'erreur est affiché et la saisie réeffectuée. Puis les paramètres globaux sont demandés, c'est-à-dire la couleur du fond de l'écran (16 choix possibles), la *palette* de couleurs qui sera utilisée pour l'avant-plan (4 palettes envisageables) et enfin le nombre d'exécutions différentes que l'utilisateur désirera voir figurer simultanément sur un même écran.

Plusieurs dessins possibles simultanément

En fonction de cette dernière réponse, un bouclage sur l'écran suivant sera effectué pour chaque exécution, afin de rendre un dessin complètement indépendant des autres. C'est ainsi qu'un tracé pourra être défini à l'intérieur d'une fenêtre bien précise, pouvant recouvrir l'intégralité de l'écran ou bien n'importe quelle surface à l'intérieur de ce dernier. La couleur du fond de la fenêtre est demandée, car elle peut être différente de celle du reste de l'écran. Puis les frontières gauche, droite, haute et basse de la fenêtre, seront entrées.

Ensuite la couleur affectée au tracé lui-même, au sein de la palette, est paramétrable. Il est également possible de désirer tracer un cadre matérialisant cette fenêtre, lequel possèdera de même sa couleur propre. Enfin les paramètres définissant le point de vue de l'observateur, par rapport à l'objet à tracer, seront entrés: **RHO**, **THETA** et **PHI**. Toutes ces données sont mémorisées, afin d'être réintroduites au moment adéquat.

Optimisation

La procédure « LectFic », quant à elle, se charge de lire et d'interpréter le fichier contenant les données de l'objet à tracer. Comme il a été indiqué ci-dessus, les nombres correspondants à des lignes de *DATA* du Basic sont entrés sans contrainte, les uns à la suite des autres. Les différentes lignes doivent simplement être terminées par un « ; ». Afin d'optimiser les temps de traitements, la lecture de ce fichier ne sera effectuée qu'une seule fois, ses éléments étant stockés dans un tableau. C'est ce dernier qui sera parcouru séquentiellement par la suite, au niveau de chaque tracé.

Enfin la procédure « Traces » effectue les différents tracés, en fonction de ce qui a été demandé en début de programme. Sa structure est tout à fait semblable à celle qui a été définie au niveau des programmes en Basic, mis à part les quelques adaptations inhérentes au langage Pascal. Quelques procédures, de niveau inférieur, participent à l'élaboration des tracés. Celle intitulée « Environ » est chargée de restituer l'environnement de chaque dessin au moment opportun.

Le Pascal est plus « parlant » que le Basic.

La procédure « Initialisations » effectue une fois pour toutes certains calculs répétitifs. La procédure « Cadre » est chargée de tracer les différents encadrés des écrans de saisie. Deux procédures, bizarrement nommées « TroisCentDix » et « TroisCentCinquante », font simplement référence aux lignes du programme écrit en Basic: **310** et **350**. Les quelques autres, relatives aux recherches portant sur la fenêtre de vision, les arcs, les sommets et les tangentes, indiquent clairement leur fonction. De même que « FicExist » et « PremLect ». Chaque tracé est également précédé de certains calculs spécifiques, effectués par les procédures « Parametres », « PreparEcran » et « Trace _ Cadres ».

Un progiciel complet

En fin d'exécution, une sonnerie indiquera l'aboutissement du travail. L'utilisation du programme est très simple, puisque l'utilisateur est guidé par une succession de *menus*, au sein desquels il lui suffira de répondre aux questions posées. Des sécurités et des contrôles jalonnent les différents écrans. Lorsqu'au moment de donner le nom d'un fichier de données à exécuter, l'utilisateur désire abandonner le traitement, il lui suffit d'entrer simplement la lettre « **Q** ». De même, en fin de travail, le fait de taper la lettre « **F** » terminera définitivement le programme et provoquera un retour propre sous **DOS**. Il ne reste plus qu'à admirer les résultats obtenus.

```
PROGRAM CAO;

{$I Graph.p}

{               Version 2.00            Auteur : Jean-Jacques MEYER
               ┌─────────────────────────────────────────────────────┐
               │ Tracés universels d'objets en trois dimensions      │
               └─────────────────────────────────────────────────────┘
}

CONST
     fi = 24;  { nombre maximum de cotés d'un polygone représentant un cercle }
     d  = 1;    { distance }
     Curs = 55;  { alignement des messages }


TYPE
    Datas = record                                   { une ligne de commandes }
                  X         : real;
                  Y         : real;
                  Z         : real;
                  Action    : real;
                  Rayon     : real;
                  Parallele : real;
                  AngleDeb  : real;
                  AngleFin  : real;
            end;

    Memo  = record
                  Col_Fen   : integer;     { 0 / 3   }
                  Lim_G_Fen : real;        { 0 / 317 }
                  Lim_D_Fen : real;        { 2 / 319 }
                  Lim_H_Fen : real;        { 0 / 197 }
                  Lim_B_Fen : real;        { 2 / 199 }
                  Col_Trace : integer;     { 0 / 3   }
                  Cadre     : char;        { O / N   }
                  Col_Cadre : integer;     { 0 / 3   }
                  Rho       : real;
                  Theta     : real;
                  Phi       : real;
            end;

    Affich = record
                   Texte : string [80];
                   Y     : integer;
             end;

    Nom    = string [30];


VAR
   Travail        : array [1..10] of Memo;
   HGx, HGy       : integer;               { coins des cadres }
   BDx, BDy       : integer;
   Execut         : integer;               { nombre de dessins }
   NumEcr         : integer;
   Ecran          : array [1..15] of Affich;
   Nom_Fic        : string [30];
   mx, my, by     : real;                  { coordonnées max. de la fenêtre }
   X1, X2, Y1, Y2 : real;
   xd, yd, np     : real;
   xx, yy, zz     : real;
   dx, dy, dz     : real;
   xr, nq, nr, ab : real;
   rp, lk, mk, re : real;
   kk, uv         : real;
   pt, qt, ru, rv : real;
```

```
   Color, Num      : integer;                         { définitions de l'écran }
   CoulEcr         : integer;
   CoulFond        : integer;
   CoulCadre       : integer;
   Car, Rep        : char;
   Nombre          : real;              { un nombre d'une ligne de commandes }
   Deci            : integer;                           { sa partie décimale }
   Top_Deci        : boolean;
   Top_Moins       : boolean;
   Top_Ok          : boolean;
   Fic_Donnees     : text;                   { fichier contenant les DATAS }
   Ind             : real;                                        { indices }
   I, J, K, L, JJ  : real;
   M, N            : integer;
   Rang            : real;
   xa, ya, xb, yb  : array [0..fi] of real;                   { sauvegardes }
   ze              : real;
   fg, fd, fh, fb  : real;  { limites de la fenêtre, de la clôture et du cadre }
   vg, vd, vh, vb  : real;
   lg, ld, lh, lb  : real;
   jc              : real;
   theta, phi      : real;                               { point de vision }
   rho             : real;
   st, sp, tc, cp  : real;
   cs, ss, sc      : real;
   xf, yf, uu      : real;                     { coordonnées temporaires }
   cg, cd, ch, cb  : real;
   Donnees         : array [0..500] of Datas;  { sauvegarde du fichier de DATA }
   AtX, AtY        : real;              { mémorisation du dernier point tracé }
   xy              : char;               { top d'indication mode tangentes (T) }
   pp              : real;
   ff              : real;                 { nombre de sommets sur un arc }
   No_Nombre, cas  : integer;
   ox, oy, oz, ti  : real;


Procedure Check;
begin
     TextMode;
     TextColor (Yellow);
     ClrScr;
     writeln ('Ce programme ne fonctionnera qu''avec la carte graphique');
     write   ('O.K. pour continuer ? (o,n) ==> ');
     repeat
           read (Kbd, Car);
           if UpCase (Car) in ['N', #27, ^C] then
              begin
                   TextMode;
                   ClrScr;
                   TextColor (Yellow);
                   Halt;
              end;
     until UpCase (Car) = 'O';
end;  { Check }


Procedure Environ;
begin
     CoulFond := Travail[NumEcr].Col_Fen;
     lg := Travail[NumEcr].Lim_G_Fen;
     ld := Travail[NumEcr].Lim_D_Fen;
     lh := Travail[NumEcr].Lim_H_Fen;
     lb := Travail[NumEcr].Lim_B_Fen;
     Color := Travail[NumEcr].Col_Trace;
     Rho := Travail[NumEcr].Rho;
     Theta := Travail[NumEcr].Theta;
     Phi := Travail[NumEcr].Phi;
end;  { Environ }
```

```
Procedure Initialisations;
begin
     fg :=  1E20;                              { limites de vision }
     fd := -1E20;
     fh := fg;
     fb := fd;
     jc := pi / 180;
     theta := theta * jc;
     phi := phi * jc;
     st := sin (theta);
     sp := sin (phi);
     tc := cos (theta);
     cp := cos (phi);
     cs := tc * sp;
     ss := st * sp;
     kk := tc * cp;
     sc := st * cp;
end;  { Initialisations }


Procedure Cadre (CoinHG_X, CoinHG_Y, CoinBD_X, CoinBD_Y: integer);
var
   I : integer;

begin
     GotoXY (CoinHG_X, CoinHG_Y);
     write (chr (218));
     for I := (CoinHG_X + 1) to (CoinBD_X - 1) do
         write (chr (196));
     write (chr (191));
     for I := (CoinHG_Y + 1) to (CoinBD_Y - 1) do
         begin
              GotoXY (CoinHG_X, I);  write (chr(179));
              GotoXY (CoinBD_X, I);  write (chr(179));
         end;
     GotoXY (CoinHG_X, CoinBD_Y);
     write (chr (192));
     for I := (CoinHG_X + 1) to (CoinBD_X - 1) do
         write (chr (196));
     write (chr (217));
end; { Cadre }


Procedure TroisCentDix;                       { recherche des coordonnées écran }
begin
     ox := (-xx * st) + (yy * tc);
     oy := (-xx * cs) - (yy * ss) + (zz * cp);
     oz := (-xx * kk) - (yy * sc) - (zz * sp) + rho;
     xd := d * ox / oz;
     yd := d * oy / oz;
end;  { TroisCentDix }


Procedure TroisCentCinquante;
begin
     xf := (xd - fg) * uu + cg;
     yf := by - ((yd - fh) * uu + ch);
end;  { TroisCentCinquante }


Procedure RechFenetre;
begin
     if xd < fg then fg := xd;
     if xd > fd then fd := xd;
     if yd < fh then fh := yd;
     if yd > fb then fb := yd;
end;  { RechFenetre }
```

```
Procedure RechNbSommets;
begin
     Cas := trunc (trunc (abs (qt - pt)) / 90);

     case Cas of

          0, 1 : ff := 0.25 * fi;

          2    : ff := 0.5 * fi;

          3    : ff := 0.75 * fi;

          4    : ff := fi;

          else   begin                                    { erreur }
                      ClrScr;
                      GotoXY (1,1);
                      writeln ('Erreur lors de la recherche du nombre de sommets. Cas = ', Cas);
                      Sound (800);
                      Delay (100);
                      NoSound;
                      Delay (200);
                      Sound (800);
                      Delay (100);
                      NoSound;
                      Halt;
                 end;  { erreur }

     end;  { du case }

     uv := (qt - pt) / ff;

end;  { RechNbSommets }


Procedure RechArcs;
begin
     case trunc (rp) of

          0, 1 : begin                              { parallèle au plan XY }
                      I := pt;
                      repeat
                            ti := I * jc;
                            xx := dx + xr * cos (ti);
                            yy := dy + xr * sin (ti);
                            TroisCentDix;
                            RechFenetre;
                            I := I + uv;
                      until I > qt;
                 end;  { 0, 1 }

          2    : begin                              { parallèle au plan YZ }
                      I := pt;
                      repeat
                            ti := I * jc;
                            yy := dy + xr * cos (ti);
                            zz := dz + xr * sin (ti);
                            TroisCentDix;
                            RechFenetre;
                            I := I + uv;
                      until I > qt;
                 end;  { 2 }

          3    : begin                              { parallèle au plan XZ }
                      I := pt;
                      repeat
                            ti := I * jc;
                            xx := dx + xr * cos (ti);
                            zz := dz + xr * sin (ti);
                            TroisCentDix;
                            RechFenetre;
                            I := I + uv;
                      until I > qt;
                 end;  { 3 }
```

```
          else   begin                                              { erreur }
                       ClrScr;
                       GotoXY (1,1);
                       writeln ('Erreur lors de la recherche des plans parallèles aux arcs. rp = ', rp);
                       Sound (800);
                       Delay (100);
                       NoSound;
                       Delay (200);
                       Sound (800);
                       Delay (100);
                       NoSound;
                       Halt;
                 end;  { erreur }

     end;  { du case }

end;  { RechArcs }


Procedure RechTangentes;
begin
     pp :=  0E0;
     nq := -1E20;
     nr :=  1E20;
     J := 1;
     repeat
          ze := xa [trunc (J)] - xb [trunc (J)];

          if trunc (ze) = 0 then
             begin
                  JJ := 1;
                  repeat
                       if xb [trunc (JJ)] > nq then
                          begin
                               lk := JJ;
                               nq := xb [trunc (JJ)];
                          end;  { du if > nq }
                       if xb [trunc (JJ)] < nr then
                          begin
                               mk := JJ;
                               nr := xb [trunc (JJ)];
                          end;  { du if > nq }
                       JJ := JJ + 1;
                  until JJ > ff;
                  J := k;
             end  { du if ze = 0 }
          else
              begin
                   ab := (yb [trunc (J)] - ya [trunc (J)]) * xb [trunc (J)] / ze + yb [trunc (J)];
                   if ab > nq then
                      begin
                           lk := J;
                           nq := ab;
                      end;  { du if > nq }
                   if ab < nr then
                      begin
                           mk := J;
                           nr := ab;
                      end;  { du if > nq }
              end;  { else ze <> 0 }
          J := J + 1;
     until J > ff;

     xx := xa [trunc (lk)];
     yy := ya [trunc (lk)];
     AtX := xx;
     AtY := yy;
     xx := xb [trunc (lk)];
     yy := yb [trunc (lk)];
     draw (trunc (AtX),trunc (AtY),  trunc (xx),trunc (yy), Color);
     xx := xa [trunc (mk)];
     yy := ya [trunc (mk)];
     AtX := xx;
     AtY := yy;
```

```
     xx := xb [trunc (mk)];
     yy := yb [trunc (mk)];
     draw (trunc (AtX),trunc (AtY),  trunc (xx),trunc (yy), Color);
end;  { RechTangentes }



Procedure Trace_ArcCercle;
begin
     dx := xx;
     dy := yy;
     dz := zz;

     case trunc (rp) of

          0, 1 : begin                              { parallèle au plan XY }
                      xx := dx + xr * cos (pt * jc);
                      yy := dy + xr * sin (pt * jc);
                      k := 0E0;
                      TroisCentDix;
                      TroisCentCinquante;
                      AtX := xf;
                      AtY := yf;

                      I := pt + uv;
                      repeat
                            ti := I * jc;
                            xx := dx + xr * cos (ti);
                            yy := dy + xr * sin (ti);
                            TroisCentDix;
                            TroisCentCinquante;
                            draw (trunc (AtX),trunc (AtY),  trunc (xf),trunc (yf), Color);
                            if xy = 'T' then
                               begin
                                    k := k + 1;
                                    xa [trunc (k)] := xf;
                                    ya [trunc (k)] := yf;
                               end;  { du if T }
                            AtX := xf;
                            AtY := yf;
                            I := I + uv;
                      until I > qt;

                      if trunc (pp) = 2 then RechTangentes;
                 end;  { 0, 1 }

          2    : begin                              { parallèle au plan YZ }
                      yy := dy + xr * cos (pt * jc);
                      zz := dz + xr * sin (pt * jc);
                      k := 0E0;
                      TroisCentDix;
                      TroisCentCinquante;
                      AtX := xf;
                      AtY := yf;

                      I := pt + uv;
                      repeat
                            ti := I * jc;
                            yy := dy + xr * cos (ti);
                            zz := dz + xr * sin (ti);
                            TroisCentDix;
                            TroisCentCinquante;
                            draw (trunc (AtX),trunc (AtY),  trunc (xf),trunc (yf), Color);
                            if xy = 'T' then
                               begin
                                    k := k + 1;
                                    xa [trunc (k)] := xf;
                                    ya [trunc (k)] := yf;
                               end;  { du if T }
                            AtX := xf;
                            AtY := yf;
                            I := I + uv;
                      until I > qt;

                      if trunc (pp) = 2 then RechTangentes;
                 end;  { 2 }
```

```
        3    : begin                                  { parallèle au plan XZ }
                  xx := dx + xr * cos (pt * jc);
                  zz := dz + xr * sin (pt * jc);
                  k := 0E0;
                  TroisCentDix;
                  TroisCentCinquante;
                  AtX := xf;
                  AtY := yf;

                  I := pt + uv;
                  repeat
                        ti := I * jc;
                        xx := dx + xr * cos (ti);
                        zz := dz + xr * sin (ti);
                        TroisCentDix;
                        TroisCentCinquante;
                        draw (trunc (AtX),trunc (AtY),  trunc (xf),trunc (yf), Color);
                        if xy = 'T' then
                           begin
                                k := k + 1;
                                xa [trunc (k)] := xf;
                                ya [trunc (k)] := yf;
                           end;  { du if T }
                        AtX := xf;
                        AtY := yf;
                        I := I + uv;
                  until I > qt;

                  if trunc (pp) = 2 then RechTangentes;
               end;  { 3 }

        else   begin                                              { erreur }
                  ClrScr;
                  GotoXY (1,1);
                  writeln ('Erreur lors du tracé d''arcs de cercles. rp = ', rp);
                  Sound (800);
                  Delay (100);
                  NoSound;
                  Delay (200);
                  Sound (800);
                  Delay (100);
                  NoSound;
                  Halt;
               end;  { erreur }
     end;  { du case }
end;  { Trace_ArcCercle }



Function FicExist (FicNom : Nom) : boolean;
begin
     assign (Fic_Donnees, FicNom);
     {$I-}
     reset (Fic_Donnees);
     {$I+}
     FicExist := (IOresult = 0);
end;  { FicExist }



Procedure LectFic;
begin
     Rang := 1;                    { RAZ de la sauvegarde du fichier de DATA }
     repeat
           Donnees [trunc (Rang)].X := 0E0;
           Donnees [trunc (Rang)].Y := 0E0;
           Donnees [trunc (Rang)].Z := 0E0;
           Donnees [trunc (Rang)].Action := 0E0;
           Donnees [trunc (Rang)].Rayon := 0E0;
           Donnees [trunc (Rang)].Parallele := 0E0;
           Donnees [trunc (Rang)].AngleDeb := 0E0;
           Donnees [trunc (Rang)].AngleFin := 0E0;
           Rang := Rang + 1;
     until Rang > 500;
```

```
Rang := 0E0;
while not eof (Fic_Donnees) do
      begin
           Car := ' ';
           No_Nombre := 0;
           Rang := Rang + 1;
           while (Car <> ';') and (not eof (Fic_Donnees)) do
                 begin
                      Top_Moins := false;
                      read (Fic_Donnees, Car);
                      Top_Deci := false;
                      while Car = ' ' do read (Fic_Donnees, Car);
                      if Car in ['0'..'9'] + ['.'] + ['-'] then
                         begin
                              Nombre := 0E0;
                              Deci := 1;
                              if Car = '-' then
                                 begin
                                      Top_Moins := true;
                                      read (Fic_Donnees, Car);
                                 end;  { - }
                              if (Car in ['0'..'9']) and (Top_Deci = false) then
                                 begin
                                      while Car in ['0'..'9'] do
                                            begin
                                                 Nombre := Nombre * 10 + ord (Car) - ord ('0');
                                                 read (Fic_Donnees, Car);
                                            end;  { nombre }
                                 end;  { chiffre }
                              if Car = '.' then
                                 begin
                                      Top_Deci := true;
                                      read (Fic_Donnees, Car);
                                      if Car in ['0'..'9'] then
                                         begin
                                              while Car in ['0'..'9'] do
                                                    begin
                                                         Deci := Deci * 10;
                                                         Nombre := Nombre + ((ord (Car) - ord ('0')) / Deci);
                                                         read (Fic_Donnees, Car);
                                                    end;  { décimales }
                                         end;  { nombre reel }
                                 end;  { . }
                              if Top_Moins = true then
                                 begin
                                      Nombre := Nombre * -1;
                                      Top_moins := false;
                                 end;  { - }
                              No_Nombre := No_Nombre + 1; { position du nombre }

                              if No_Nombre = 1 then
                                      Donnees [trunc (Rang)].X := Nombre;

                              if No_Nombre = 2 then
                                      Donnees [trunc (Rang)].Y := Nombre;

                              if No_Nombre = 3 then
                                      Donnees [trunc (Rang)].Z := Nombre;

                              if No_Nombre = 4 then
                                      Donnees [trunc (Rang)].Action := trunc (Nombre);

                              if No_Nombre = 5 then
                                      Donnees [trunc (Rang)].Rayon := Nombre;

                              if No_Nombre = 6 then
                                      Donnees [trunc (Rang)].Parallele := trunc (Nombre);

                              if No_Nombre = 7 then
                                      Donnees [trunc (Rang)].AngleDeb := Nombre;

                              if No_Nombre = 8 then
                                      Donnees [trunc (Rang)].AngleFin := Nombre;

                         end;  { chiffre . ou - }

                 end;  { fin d'une ligne }
```

```
              readln (Fic_Donnees);
              No_Nombre := 0;

          end;  { fin du fichier }

     Rang := Rang + 1;
     Donnees [trunc (Rang)].Action := 4;                    { fin de la table }

     close (Fic_Donnees);
end;  { LectFic }



Procedure PremLect;
begin
     Rang := 1;
     repeat
          xx := Donnees [trunc (Rang)].X;
          yy := Donnees [trunc (Rang)].Y;
          zz := Donnees [trunc (Rang)].Z;
          np := Donnees [trunc (Rang)].Action;

          case trunc (np) of

              0, 1 : begin                                          { droite }
                          TroisCentDix;
                          RechFenetre;
                     end;  { Nombre = 0, 1 }

              2    : begin                                    { arc de cercle }
                          xr := Donnees [trunc (Rang)].Rayon;
                          rp := Donnees [trunc (Rang)].Parallele;
                          pt := Donnees [trunc (Rang)].AngleDeb;
                          qt := Donnees [trunc (Rang)].AngleFin;
                          RechNbSommets;
                          RechArcs;
                     end;  { Nombre = 2 }

              3    : begin                                          { cercle }
                          xr := Donnees [trunc (Rang)].Rayon;
                          rp := Donnees [trunc (Rang)].Parallele;
                          pt := 0E0;
                          qt := 360;
                          RechNbSommets;
                          RechArcs;
                     end;  { Nombre = 3 }

              4    : ;                                    { fin de la table }

              else   begin                                          { erreur }
                          ClrScr;
                          GotoXY (1,1);
                          writeln ('Erreur au niveau des données (action > 3)');
                          Sound (800);
                          Delay (100);
                          NoSound;
                          Delay (200);
                          Sound (800);
                          Delay (100);
                          NoSound;
                          Halt;
                     end;  { erreur }

          end;  { du case }

          Rang := Rang + 1;
     until Donnees [trunc (Rang)].Action = 4;

     ru := (cd - cg) / (fd - fg);
     rv := (cb - ch) / (fb - fh);
     uu := ru;
     if rv < ru then uu := rv;
end;  { PremLect }
```

```
Procedure Parametres;
begin
     mx := int (abs (lg - ld));
     my := int (abs (lh - lb));
     by := my - 1;
     cg := 0E0;
     cd := mx;
     ch := 0E0;
     cb := my;
     re := ch;                            { pour type écran }
     ch := my - 1 - cb;
     cb := my - 1 - re;
end;  { Parametres }



Procedure PreparEcran;
begin
     GraphWindow (trunc (lg),trunc (lh), trunc (ld),trunc (lb));    { fenetre }
     FillScreen (CoulFond);                  { couleur du fond de la fenetre }
end;  { PreparEcran }



Procedure Trace_Cadres;
begin
  NumEcr := 1;                         { tracé des cadres éventuels }
  repeat
        if UpCase (Travail[NumEcr].Cadre) = 'O' then
           begin
                CoulCadre := Travail[NumEcr].Col_Cadre;
                vg := Travail[NumEcr].Lim_G_Fen - 3;
                vd := Travail[NumEcr].Lim_D_Fen + 3;
                vh := Travail[NumEcr].Lim_H_Fen - 3;
                vb := Travail[NumEcr].Lim_B_Fen + 3;
                Draw (trunc (vg),trunc (vh), trunc (vd),trunc (vh), CoulCadre);
                Draw (trunc (vd),trunc (vh), trunc (vd),trunc (vb), CoulCadre);
                Draw (trunc (vd),trunc (vb), trunc (vg),trunc (vb), CoulCadre);
                Draw (trunc (vg),trunc (vb), trunc (vg),trunc (vh), CoulCadre);
           end;  { cadre à tracer }
        NumEcr := NumEcr + 1;
  until NumEcr > Execut;
end;  { Trace_Cadres }



Procedure Traces;
begin
  GraphColorMode;             { 320 x 200 pixels, 4 palettes de 4 couleurs }
  GraphBackGround (CoulEcr);
  TextColor (white);
  Palette (Num);
  ColorTable (0,1,2,3);                { pas de modification des couleurs }
  Trace_Cadres;

  NumEcr := 1;
  repeat
     Environ;
     Parametres;
     Initialisations;
     PremLect;
     PreparEcran;
```

```
{ ################ lecture de la table, pour les tracés ############## }
Rang := 1;
repeat
      xx := Donnees [trunc (Rang)].X;
      yy := Donnees [trunc (Rang)].Y;
      zz := Donnees [trunc (Rang)].Z;
      np := Donnees [trunc (Rang)].Action;
      TroisCentDix;

      case trunc (np) of

           0 : begin               { début de tracé de segment de droite }
                    TroisCentCinquante;
                    AtX := xf;
                    AtY := yf;
               end;  { 0 }

           1 : begin               { fin de tracé de segment de droite }
                    TroisCentCinquante;
                    draw (trunc (AtX),trunc (AtY),  trunc (xf),trunc (yf), Color);
                    AtX := xf;
                    AtY := yf;
               end;  { 1 }

           2 : begin                                   { arc de cercle }
                    xr := Donnees [trunc (Rang)].Rayon;
                    rp := Donnees [trunc (Rang)].Parallele;
                    pt := Donnees [trunc (Rang)].AngleDeb;
                    qt := Donnees [trunc (Rang)].AngleFin;
                    RechNbSommets;
                    Trace_ArcCercle;
               end;  { 2 }

           3 : begin                                          { cercle }
                    xr := Donnees [trunc (Rang)].Rayon;
                    rp := Donnees [trunc (Rang)].Parallele;
                    xy := 'T';
                    pp := pp + 1;
                    if pp <> 1 then
                       begin
                            I := 1;
                            repeat
                                     xb [trunc (I)] := xa [trunc (I)];
                                     yb [trunc (I)] := ya [trunc (I)];
                                     I := I + 1;
                            until I > ff;
                       end;  { pp <> 1 }
                    pt := 0E0;
                    qt := 360;
                    RechNbSommets;
                    Trace_ArcCercle;
                    xy := ' ';
               end;  { 3 }

           4 : ;                                                 { fin }

           else  begin                                        { erreur }
                      ClrScr;
                      GotoXY (1,1);
                      writeln ('Erreur lors de la recherche de l''action, pour les tracés. np = ', np);
                      Sound (800);
                      Delay (100);
                      NoSound;
                      Delay (200);
                      Sound (800);
                      Delay (100);
                      NoSound;
                      Halt;
                 end;  { erreur }

      end;  { du case }

      Rang := Rang + 1;
until Donnees [trunc (Rang)].Action = 4;             { fin de la table }
```

```
    NumEcr := NumEcr + 1;
  until NumEcr > Execut;

  Sound (440);                                          { fin du tracé }
  Delay (100);
  NoSound;
  Delay (200);
  Sound (600);
  Delay (100);
  NoSound;

  repeat
       read (Kbd, Car);
  until UpCase (Car) = 'F';
end;  { Traces }



Procedure Entrees;
begin
     Ecran[1].Texte := 'Nom du fichier de données à exécuter :';
     Ecran[1].Y := 9;
     Ecran[2].Texte := 'Couleur de l''écran [0 - 15]  (1) :';
     Ecran[2].Y := 11;
     Ecran[3].Texte := 'Numéro de la palette [0 - 3]  (3) :';
     Ecran[3].Y := 13;
     Ecran[4].Texte := 'Nombre d''exécutions différentes [1 - 10]  (1) :';
     Ecran[4].Y := 15;

     Ecran[5].Texte := 'Couleur fenêtre [0 - 3]  (1) :';
     Ecran[5].Y := 6;
     Ecran[6].Texte := 'Limite GAUCHE fenêtre [319]     (4) :';
     Ecran[6].Y := 8;
     Ecran[7].Texte := 'Limite DROITE fenêtre [319]  (315) :';
     Ecran[7].Y := 9;
     Ecran[8].Texte := 'Limite HAUTE  fenêtre [199]     (4) :';
     Ecran[8].Y := 10;
     Ecran[9].Texte := 'Limite BASSE  fenêtre [199]  (195) :';
     Ecran[9].Y := 11;
     Ecran[10].Texte := 'Couleur du tracé [0 - 3]  (3) :';
     Ecran[10].Y := 13;
     Ecran[11].Texte := 'Cadre desiré  (O/N) :';
     Ecran[11].Y := 14;
     Ecran[12].Texte := 'Couleur du cadre [0 - 3]  (2) :';
     Ecran[12].Y := 15;
     Ecran[13].Texte := 'Distance de l''observateur RHO  (400) :';
     Ecran[13].Y := 17;
     Ecran[14].Texte := 'Angle THETA en degrés  (60) :';
     Ecran[14].Y := 18;
     Ecran[15].Texte := 'Angle PHI en degrés  (20) :';
     Ecran[15].Y := 19;

     ClrScr;
     NormVideo;
     HGx := 1;  HGy := 1;
     BDx := 80; BDy := 4;
     Cadre (HGx, HGy, BDx, BDy);
     HGx := 1;  HGy := 22;
     BDx := 80; BDy := 24;
     Cadre (HGx, HGy, BDx, BDy);

     LowVideo;
     GotoXY (2,2);
     write ('                                   E N T R E E  des  D O N N E E S');
     GotoXY (2,3);
     write ('                                                               Niveau Global');

     GotoXY (Curs - Length (Ecran[1].Texte) - 1, Ecran[1].Y);
     write  (Ecran[1].Texte);
     GotoXY (Curs, Ecran[1].Y);
     readln (Nom_Fic);
     Top_Ok := true;
```

```
if not FicExist (Nom_Fic) then                  { erreur de fichier }
        Top_Ok := false;
if Top_Ok = false then
   begin
        repeat
              Sound (800);                      { bip sonore }
              Delay (100);
              NoSound;
              GotoXY (3,23);
              write ('Fichier inexistant');
              for M := 1 to 32000 do
                  for N := 1 to 2 do;
              GotoXY (3,23);
              write ('                  ');
              GotoXY (Curs, Ecran[1].Y);        { ressaisie }
              ClrEol;
              readln (Nom_Fic);
              Top_Ok := true;
              if not FicExist (Nom_Fic) then    { erreur de fichier }
                 Top_Ok := false;
        until (Top_Ok = true) or (UpCase (Nom_Fic[1]) = 'Q');
        if UpCase (Nom_Fic[1]) = 'Q' then
           begin
                TextMode;
                ClrScr;
                TextColor (Yellow);
                Halt;
           end;  { fin }
   end;  { erreur de fichier }

GotoXY (Curs - Length (Ecran[2].Texte) - 1, Ecran[2].Y);
write  (Ecran[2].Texte);
GotoXY (Curs, Ecran[2].Y);
readln (CoulEcr);
if CoulEcr < 0 then
   CoulEcr := 0;
if CoulEcr > 15 then
   CoulEcr := 15;

GotoXY (Curs - Length (Ecran[3].Texte) - 1, Ecran[3].Y);
write  (Ecran[3].Texte);
GotoXY (Curs, Ecran[3].Y);
readln (Num);
if Num < 0 then
   Num := 0;
if Num > 3 then
   Num := 3;

GotoXY (Curs - Length (Ecran[4].Texte) - 1, Ecran[4].Y);
write  (Ecran[4].Texte);
GotoXY (Curs, Ecran[4].Y);
readln (Execut);
if Execut < 1 then
   Execut := 1;
if Execut > 10 then
   Execut := 10;

GotoXY (3,23);
write ('Appuyer sur une touche pour continuer ');
repeat until KeyPressed;

ClrScr;
NormVideo;
HGx := 1;  HGy := 1;
BDx := 80; BDy := 4;
Cadre (HGx, HGy, BDx, BDy);
HGx := 1;  HGy := 22;
BDx := 80; BDy := 24;
Cadre (HGx, HGy, BDx, BDy);
```

```
LowVideo;
GotoXY (2,2);
writeln ('                                    E N T R E E  des  D O N N E E S');
GotoXY (2,3);
writeln ('                                              Pour l''exécution Nº ');

NumEcr := 1;                              { bouclage par écran }
repeat
      GotoXY (3,23);
      write ('                                    ');
      GotoXY (78,3);
      write (NumEcr);

      GotoXY (Curs - Length (Ecran[5].Texte) - 1, Ecran[5].Y);
      write  (Ecran[5].Texte);
      GotoXY (Curs, Ecran[5].Y);  ClrEol;
      readln (Travail[NumEcr].Col_Fen);
      if Travail[NumEcr].Col_Fen < 0 then
         Travail[NumEcr].Col_Fen := 0;
      if Travail[NumEcr].Col_Fen > 3 then
         Travail[NumEcr].Col_Fen := 3;

      GotoXY (Curs - Length (Ecran[6].Texte) - 1, Ecran[6].Y);
      write  (Ecran[6].Texte);
      GotoXY (Curs, Ecran[6].Y);  ClrEol;
      readln (Travail[NumEcr].Lim_G_Fen);
      if Travail[NumEcr].Lim_G_Fen < 0 then
         Travail[NumEcr].Lim_G_Fen := 0;
      if Travail[NumEcr].Lim_G_Fen > 319 then
         Travail[NumEcr].Lim_G_Fen := 319;

      GotoXY (Curs - Length (Ecran[7].Texte) - 1, Ecran[7].Y);
      write  (Ecran[7].Texte);
      GotoXY (Curs, Ecran[7].Y);  ClrEol;
      readln (Travail[NumEcr].Lim_D_Fen);
      if Travail[NumEcr].Lim_D_Fen < 0 then
         Travail[NumEcr].Lim_D_Fen := 0;
      if Travail[NumEcr].Lim_D_Fen > 319 then
         Travail[NumEcr].Lim_D_Fen := 319;

      GotoXY (Curs - Length (Ecran[8].Texte) - 1, Ecran[8].Y);
      write  (Ecran[8].Texte);
      GotoXY (Curs, Ecran[8].Y);  ClrEol;
      readln (Travail[NumEcr].Lim_H_Fen);
      if Travail[NumEcr].Lim_H_Fen < 0 then
         Travail[NumEcr].Lim_H_Fen := 0;
      if Travail[NumEcr].Lim_H_Fen > 199 then
         Travail[NumEcr].Lim_H_Fen := 199;

      GotoXY (Curs - Length (Ecran[9].Texte) - 1, Ecran[9].Y);
      write  (Ecran[9].Texte);
      GotoXY (Curs, Ecran[9].Y);  ClrEol;
      readln (Travail[NumEcr].Lim_B_Fen);
      if Travail[NumEcr].Lim_B_Fen < 0 then
         Travail[NumEcr].Lim_B_Fen := 0;
      if Travail[NumEcr].Lim_B_Fen > 199 then
         Travail[NumEcr].Lim_B_Fen := 199;

      GotoXY (Curs - Length (Ecran[10].Texte) - 1, Ecran[10].Y);
      write  (Ecran[10].Texte);
      GotoXY (Curs, Ecran[10].Y);  ClrEol;
      readln (Travail[NumEcr].Col_Trace);
      if Travail[NumEcr].Col_Trace < 0 then
         Travail[NumEcr].Col_Trace := 0;
      if Travail[NumEcr].Col_Trace > 3 then
         Travail[NumEcr].Col_Trace := 3;
```

```
            GotoXY (Curs - Length (Ecran[11].Texte) - 1, Ecran[11].Y);
            write  (Ecran[11].Texte);
            GotoXY (Curs, Ecran[11].Y);  ClrEol;
            readln (Travail[NumEcr].Cadre);
            if (UpCase (Travail[NumEcr].Cadre) <> 'O') and (UpCase (Travail[NumEcr].Cadre) <> 'N') then
               Travail[NumEcr].Cadre := 'N';
            if UpCase (Travail[NumEcr].Cadre) = 'O' then
               begin
                   GotoXY (Curs - Length (Ecran[12].Texte) - 1, Ecran[12].Y);
                   write  (Ecran[12].Texte);
                   GotoXY (Curs, Ecran[12].Y);  ClrEol;
                   readln (Travail[NumEcr].Col_Cadre);
                   if Travail[NumEcr].Col_Cadre < 0 then
                      Travail[NumEcr].Col_Cadre := 0;
                   if Travail[NumEcr].Col_Cadre > 3 then
                      Travail[NumEcr].Col_Cadre := 3;
               end  { if O }
            else
                GotoXY (Curs, Ecran[12].Y); ClrEol;

            GotoXY (Curs - Length (Ecran[13].Texte) - 1, Ecran[13].Y);
            write  (Ecran[13].Texte);
            GotoXY (Curs, Ecran[13].Y);  ClrEol;
            readln (Travail[NumEcr].Rho);

            GotoXY (Curs - Length (Ecran[14].Texte) - 1, Ecran[14].Y);
            write  (Ecran[14].Texte);
            GotoXY (Curs, Ecran[14].Y);  ClrEol;
            readln (Travail[NumEcr].Theta);

            GotoXY (Curs - Length (Ecran[15].Texte) - 1, Ecran[15].Y);
            write  (Ecran[15].Texte);
            GotoXY (Curs, Ecran[15].Y);  ClrEol;
            readln (Travail[NumEcr].Phi);

            GotoXY (3,23);
            write ('Appuyer sur une touche pour continuer ');
            repeat until KeyPressed;

            NumEcr := NumEcr + 1;
      until NumEcr > Execut;

      ClrScr;
      GotoXY (1,1);
      NormVideo;
      write ('[');
      LowVideo;  write ('F');
      NormVideo;  writeln ('] terminera le programme, en fin de tracés.');
      GotoXY (5,3);
      writeln ('Calculs en cours ...');
      LowVideo;
      for M := 1 to 32000 do              { temporisation }
          for N := 1 to 2 do;
end;  { Entrees }


BEGIN   { PROGRAMME PRINCIPAL }
      Check;
      Entrees;
      LectFic;
      Traces;
      TextMode;
      ClrScr;
      TextColor (Yellow);
END.
```

Exemples de fichiers de données :

Mécanique

```
PIECE UN
0 115 0 3 4 3;
0 100 0 3 4 3;
0 115 0 3 5 3;
0 110 0 3 9 3;
0 110 0 3 9 3;
0 100 0 3 9 3;

PIECE DEUX
0 90 0 3 20 3;
0 80 0 3 18 3;
0 80 0 3 18 3;
0 70 0 3  4 3;
0 90 0 3 22 3;
0 80 0 3 20 3;
0 80 0 3 20 3;
0 70 0 3 12 3;
```

```
PIECE TROIS
  0 60 0 3 22 3;
  0 55 0 3 27 3;
  0 55 0 3 27 3;
  0 45 0 3 20 3;
  0 45 0 3 20 3;
  0 20 0 3 50 3;
  0 20 0 3 50 3;
  0 15 0 3 50 3;
  0 60 0 3 15 3;
  0 15 0 3 15 3;
 40 20 0 3  3 3;
 40 15 0 3  3 3;
-40 20 0 3  3 3;
-40 15 0 3  3 3;
```

```
PIECE QUATRE
0   5 0 3 10 3;
0   3 0 3 10 3;
0   3 0 3 10 3;
0 -10 0 3 15 3;
0   5 0 3  4 3;
0   3 0 3  4 3;
0   3 0 3  8 3;
0 -10 0 3 13 3;

PIECE CINQ
0 -20 0 3  4 3;
0 -22 0 3  4 3;
0 -20 0 3 15 3;
0 -22 0 3 15 3;
```

```
PIECE SIX
0 -30 0 3 13 3;
0 -35 0 3 13 3;
0 -30 0 3 15 3;
0 -35 0 3 15 3;

PIECE SEPT
0 -45 0 3  4 3;
0 -60 0 3  4 3;
0 -45 0 3  6 3;
0 -60 0 3  6 3;
0 -45 0 3 15 3;
0 -50 0 3 18 3;
```

```
PIECE HUIT
0  -70 0 3  6 3;
0  -75 0 3  6 3;
0  -70 0 3 25 3;
0  -75 0 3 25 3;
0  -75 0 3 20 3;
0  -77 0 3 20 3;
0  -75 0 3 23 3;
0  -77 0 3 23 3;
0  -77 0 3 20 3;
0 -110 0 3 35 3;
0  -77 0 3 25 3;
0 -110 0 3 40 3;
0 -110 0 3 22 3;
0 -115 0 3 22 3;
0 -110 0 3 45 3;
0 -115 0 3 45 3;
```

Avion

```
FUSELAGE
 45 0 0 0;
 40 0 0 1;
 40 0 0 3 1 2;
 38 0 0 3 3 2;
 38 0 0 3 3 2;
 30 0 0 3 5 2;
 30 0 0 3 5 2;
 20 0 0 3 7 2;
 20 0 0 3 7 2;
 15 0 0 3 8 2;
 15 0 0 3 8 2;
  0 0 0 3 5 2;
  0 0 0 3 5 2;
-10 0 0 3 5 2;
-10 0 0 3 5 2;
-20 0 0 3 4 2;
-20 0 0 3 4 2;
-30 0 0 3 4 2;
-30 0 0 3 4 2;
-42 0 0 3 3 2;
-42 0 0 3 3 2;
-50 0 0 3 2 2;
-50 0 0 3 2 2;
-55 0 0 3 1 2;
```

```
AILE
2.5  75 -2 2 2.5 1 0 180;
2.5  75 -4 2 2.5 1 0 180;
0   -75 -4 1;
2.5 -75 -4 2 2.5 1 180 360;
2.5 -75 -2 2 2.5 1 180 360;
0    75 -2 0;
0   -75 -2 1;
5    75 -2 0;
15    8 -2 1;
15   -8 -2 1;
5   -75 -2 1;
5    75 -4 0;
15    8 -4 1;
15   -8 -4 1;
5   -75 -4 1;
```

```
AILERON
-48.5  25  1 2 1.5 1 0 180;
-48.5  25 -1 2 1.5 1 0 180;
-50   -25 -1 1;
-48.5 -25 -1 2 1.5 1 180 360;
-48.5 -25  1 2 1.5 1 180 360;
-50    25  1 0;
-50   -25  1 1;
-47    25  1 0;
-42     3  1 1;
-42    -3  1 1;
-47   -25  1 1;
-47    25 -1 0;
-42     3 -1 1;
-42    -3 -1 1;
-47   -25 -1 1;
```

```
REACTEURS
20  27.5 -6.5 3 2 2;
-5  27.5 -6.5 3 2 2;
20 -27.5 -6.5 3 2 2;
-5 -27.5 -6.5 3 2 2;

DERIVE
-35 0 0  0;
-50 0 0  1;
-50 0 20 1;
-47 0 20 1;
-35 0 3  1;
-35 0 0  1;
```

Géométrie

```
PREMIER PLAN, HORIZONTAL
 60 -80  80 0;
-60 -80 -80 1;
-60  80 -80 1;
 60  80  80 1;
 60 -80  80 1;

DEUXIEME PLAN, VERTICAL
 60 -80 -80 0;
-60 -80  80 1;
-60  80  80 1;
 60  80 -80 1;
 60 -80 -80 1;

INTERSECTION
  0 -80   0 0;
  0  80   0 1;

PREMIERE DROITE, A DROITE
 40  70 -70 0;
-45 -20  60 1;

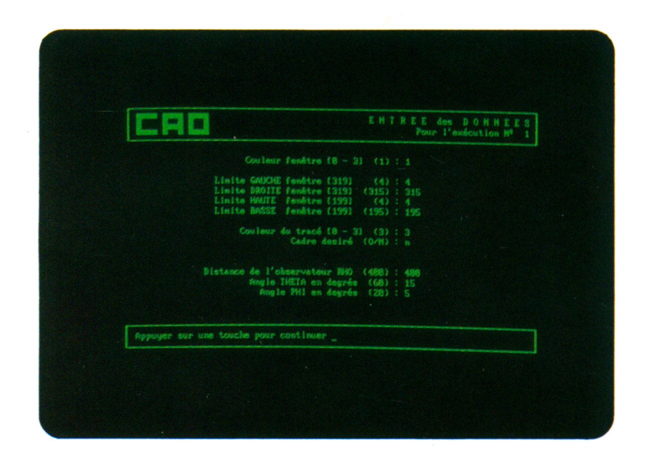
DEUXIEME DROITE, A GAUCHE
-40  70 -60 0;
 35 -70  90 1;

CYLINDRE
 32 -63  40 3 12 2;
-45 -50  55 3 10 2;

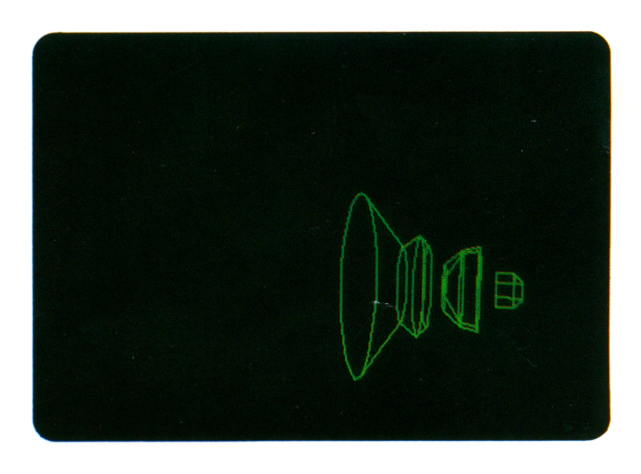
PYRAMIDE
 38   3 -70 0;
 38  35 -70 1;
 -5  20 -70 1;
 38   3 -70 1;
 22  20 -30 1;
 38  35 -70 1;
 22  20 -30 0;
 -5  20 -70 1;
```

EXEMPLES DE REALISATIONS

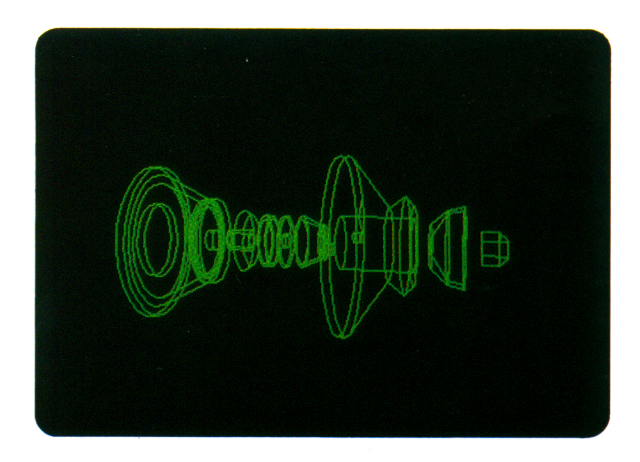
Ces photographies illustrent quelques-unes des possibilités de création graphique à l'aide des différents programmes décrits dans ce livre, notamment à l'aide du progiciel de CAO écrit en Turbo Pascal.



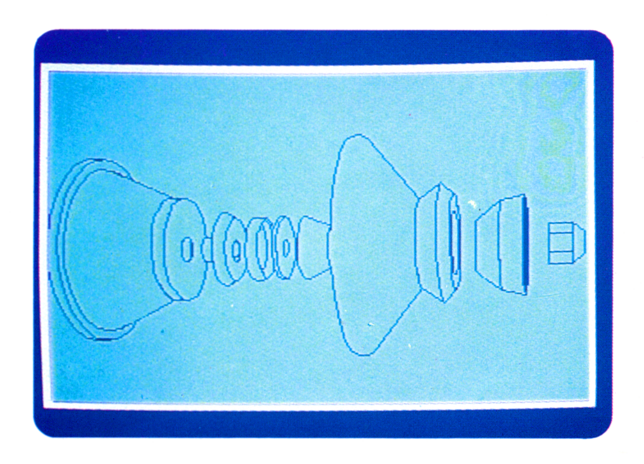
Avec ce logiciel en Turbo Pascal, dessiner rapidement en 3 D avec une perspective réelle est vraiment très aisé. Il suffit de répondre à quelques questions.



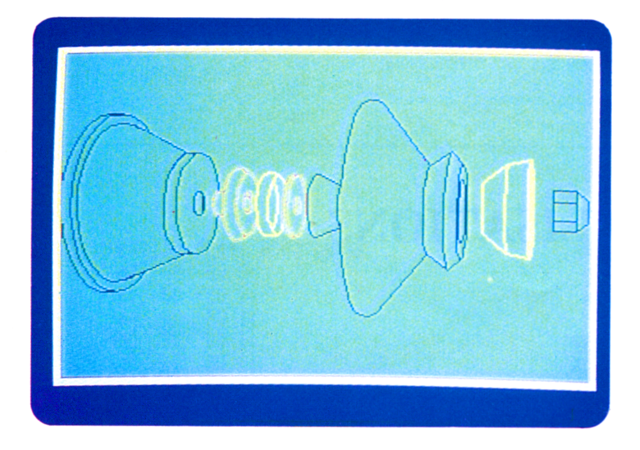
Puis le programme commence le tracé, après de savants calculs, à partir de coordonnées répertoriées dans un simple fichier ASCII.



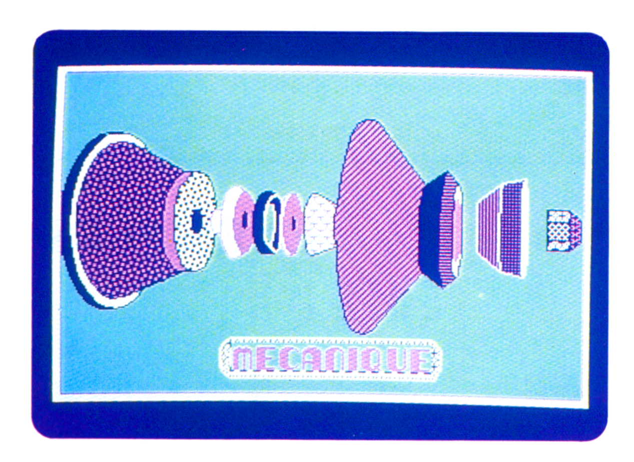
Le tracé primaire est maintenant terminé. Il peut être recommencé à volonté en modifiant la position de l'observateur par rapport à l'objet.



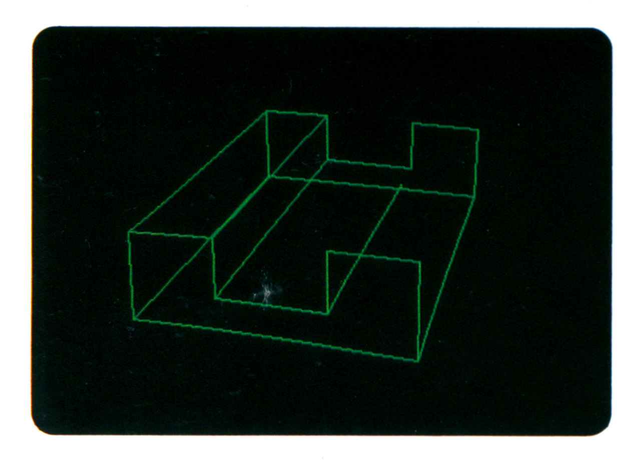
Lorsque le résultat est jugé satisfaisant, il est possible de l'améliorer en utilisant un utilitaire de dessin. Il convient d'abord d'en éliminer les surfaces cachées.



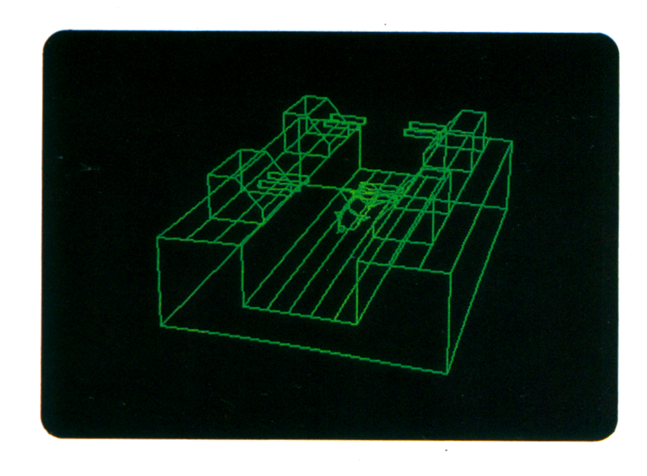
Puis de commencer à colorier les différentes composantes du dessin, avec essais successifs.



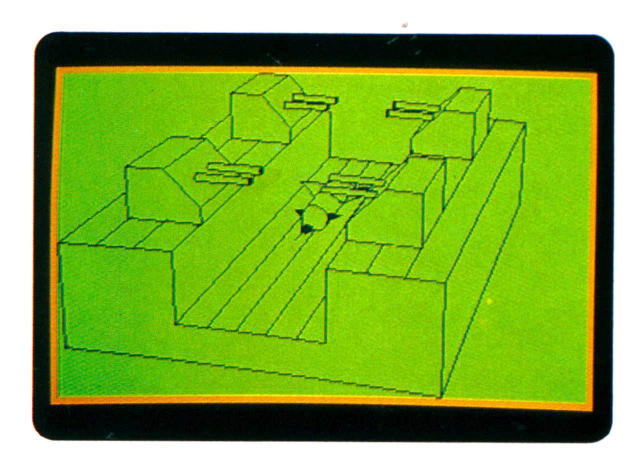
Et voici le résultat : bénéficier de la puissance d'un utilitaire de dessin tout en autorisant l'emploi des règles du dessin en perspective réelle, automatiquement.



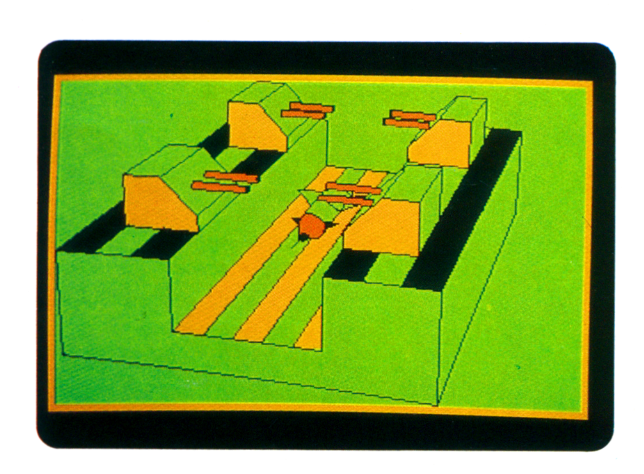
Il suffit de changer le fichier des coordonnées pour commencer à tracer un autre objet, selon le point de vue désiré.



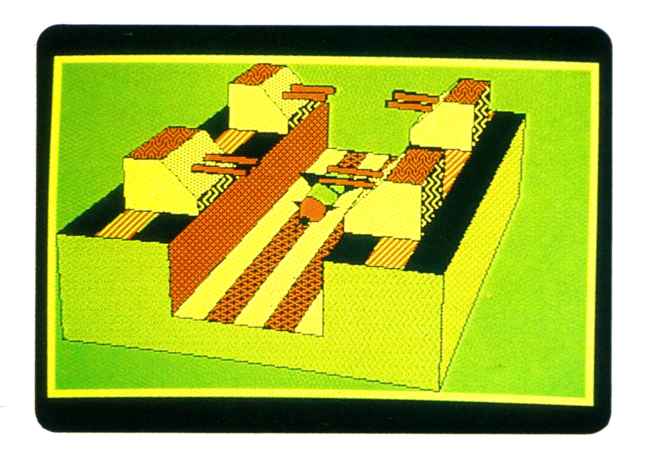
Le tracé en 3 D est maintenant terminé, avec la représentation de toutes ses lignes, ce qui peut être utile en cas de surfaces transparentes.



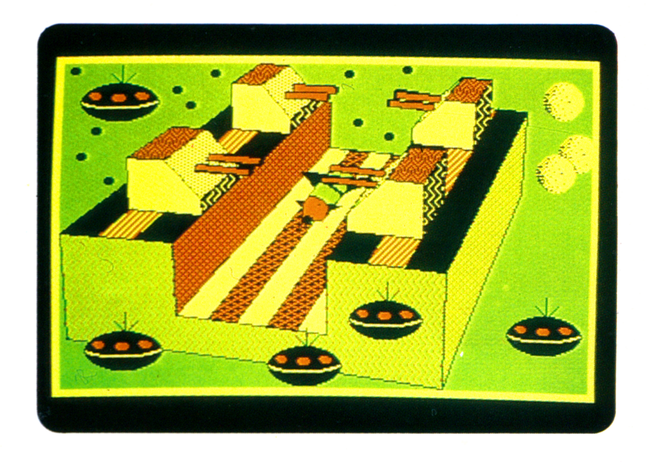
Les lignes cachées indésirables sont ensuite éliminées, par l'emploi d'un utilitaire de dessin.



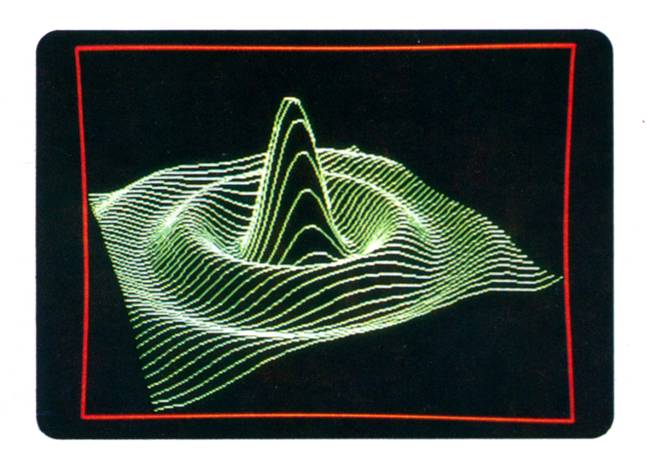
Puis les retouches et la mise en couleurs peuvent débuter.



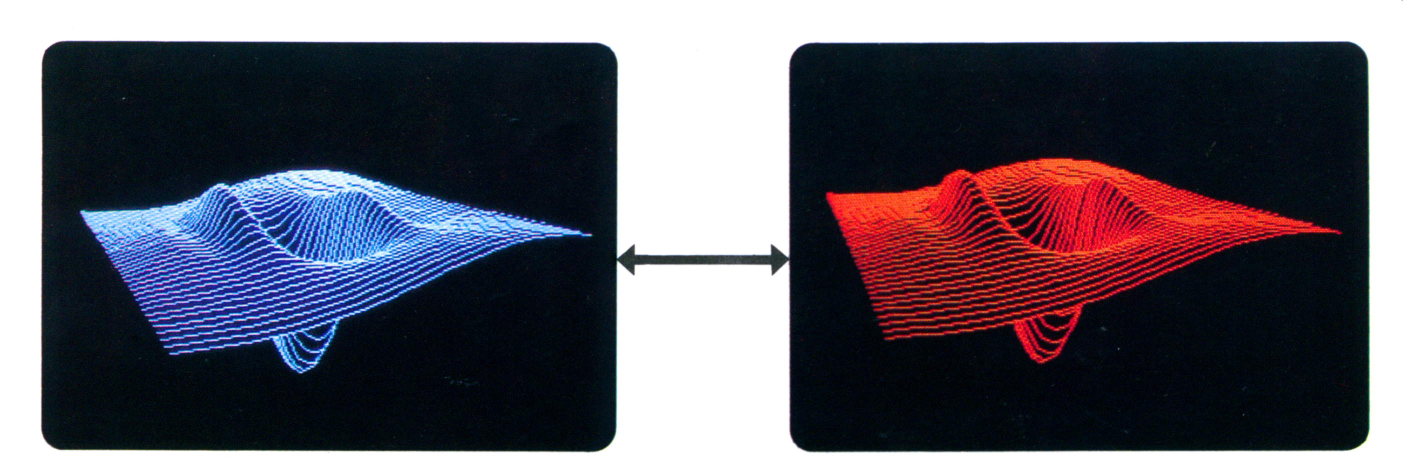
Le dessin est maintenant terminé. Différentes couleurs de fond et palettes peuvent être essayées.



Il est encore possible de le «peaufiner» davantage, en utilisant toutes les ressources de l'utilitaire de dessin.

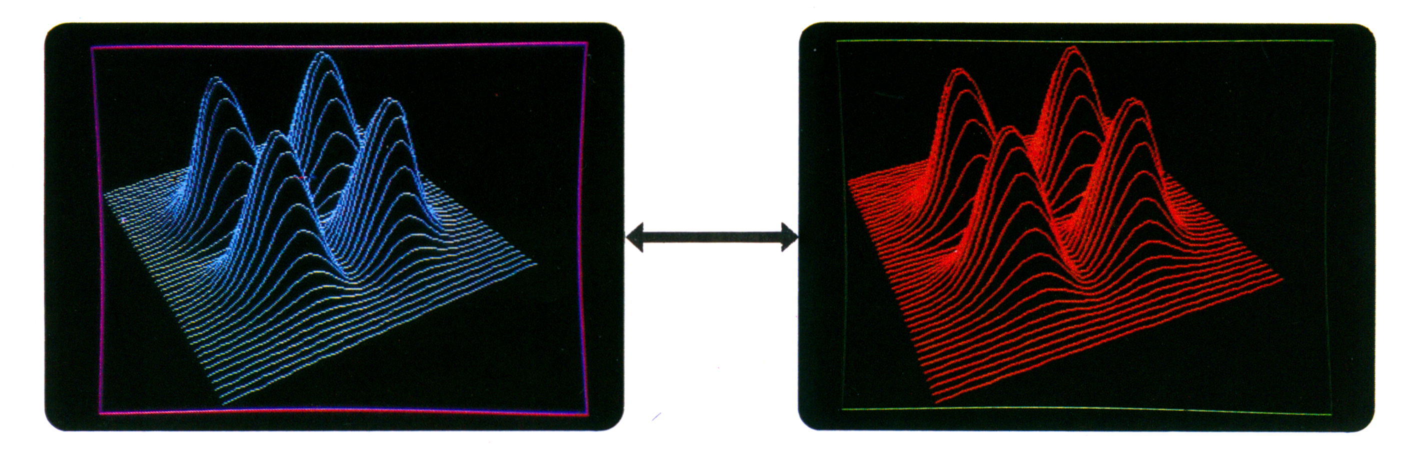


Un autre programme, écrit en Basic Microsoft portable, autorise la génération de fonctions mathématiques en 3 D, avec élimination automatique des lignes cachées, en fonction du point de vue.



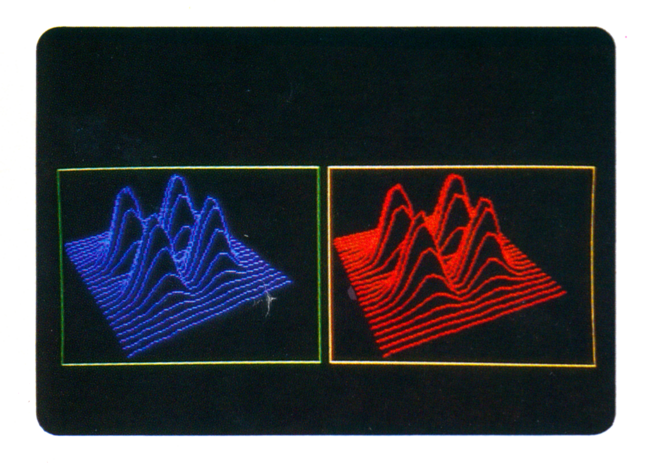
N'importe quelle fonction peut être représentée, sous un angle quelconque. Un tracé effectué en bleu pourra servir à une visualisation en relief, pour l'oeil gauche.

Le même tracé, mais effectué avec un décalage respectant les lois du relief et l'emploi de la couleur rouge, concernera l'oeil droit.

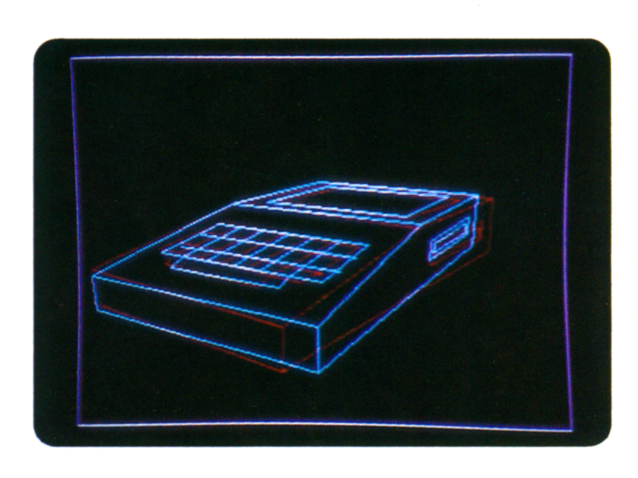


Une autre fonction mathématique, destinée à la visualisation en relief directement sur l'écran, avec des lunettes bicolores.

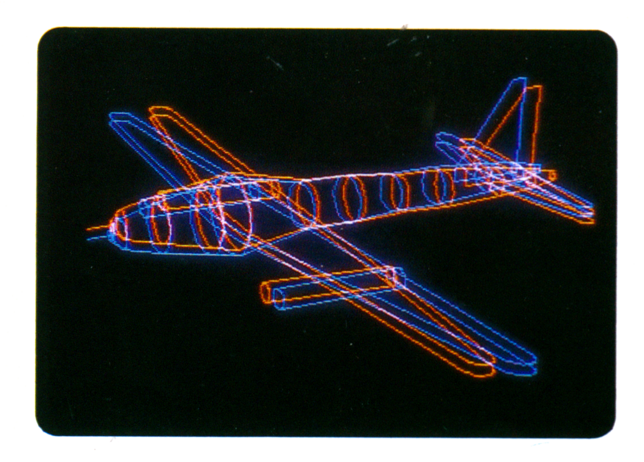
En voici la vue de droite, tracée en rouge. Des mises en couleurs sont possibles, avec d'autres procédés de visualisation en relief.



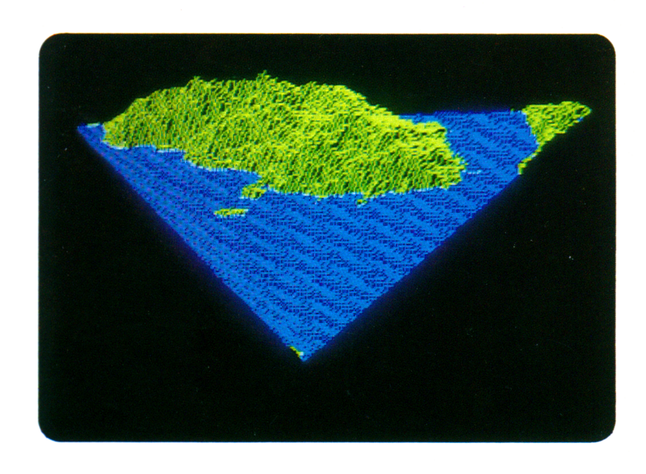
Les deux tracés nécessaires au relief peuvent aussi être effectués sur un même écran.



Il est également possible de représenter des objets concaves en 3 D et vraie perspective, avec élimination automatique des surfaces cachées, selon les mêmes principes.



Ou même des objets complexes de toutes formes, destinés à être visualisés en vrai relief, selon l'angle de vision désiré.



Enfin, un logiciel permet de générer automatiquement des paysages d'apparence réelle, respectant les lois du hasard et de la nature.

CONCLUSION

En informatique, comme en général dans la plupart des domaines, il est important de bien définir certains compromis. Il est donc nécessaire d'effectuer des choix, entre langage proche de l'utilisateur mais logiciels volumineux, facilité et universalité d'utilisation mais exécution lente et langages plus complexes et plus spécifiques, programmes moins longs mais d'exécution plus rapide.

Le but recherché était de permettre à **tous**, quel que soit le matériel possédé, de réaliser des logiciels performants dans le domaine de la **troisième dimension** d'une part et du **relief** d'autre part. Pour ces raisons, l'utilisation du langage *BASIC* s'imposait. Il est présent sur la quasi totalité des micro-ordinateurs disponibles actuellement sur le marché et son universalité est largement reconnue.

Par contre, lorsque l'on désire apporter davantage de performances à un logiciel, l'utilisation d'autres langages devient rapidement impérative. Ces derniers sont souvent moins connus et d'utilisation plus difficile, mais leur rapidité d'exécution est sans commune mesure avec celle du **Basic** *interprété.* C'est ainsi que le langage **Pascal** a été choisi comme exemple d'adaptation, à la fin de la troisième partie de ce livre.

Et plus particulièrement le **Turbo Pascal** qui, tout en conservant une certaine aisance d'utilisation, apporte cependant rapidité d'exécution, implémentation graphique et reconnaissance de plus en plus marquée en tant que standard. Il serait également possible d'utiliser le langage « **C** », qui apporterait un surcroît de rapidité, mais au prix de plus grandes complications au niveau de la programmation graphique. Enfin l'assembleur demeure encore le moyen *royal* d'obtenir le maximum de rapidité d'exécution. Par contre il est difficile à maîtriser et très spécifique à chaque micro-processeur.

Des améliorations sont encore possibles, au niveau des logiciels écrits en Pascal. Par exemple il serait intéressant de stocker un écran dans un fichier et de le rappeler par la suite, à volonté. Des logiciels comme « **PC-PAINT** » rendent possibles bien des manipulations et **Turbo Pascal** permet de gérer une grande variété de fichiers. L'utilisation de petites routines, en assembleur, permettrait de stocker des écrans à différents endroits de la mémoire, puis de les faire défiler instantanément, en vue de l'obtention directe de l'effet de **relief** à l'écran ou d'effets d'**animations**. Pour cela il n'est pas nécessaire de posséder un assembleur. **DEBUG**, sur **IBM-PC** permet de composer ces sortes de petites routines en langage machine.

Enfin, une amélioration importante consisterait à supprimer le relevé manuel des différentes cotes des dessins. Le dernier programme a pompeusement été baptisé «progiciel». Pourtant il faut reconnaître que pour mériter ce nom il devrait être modifié, en vue de lui apporter davantage de souplesse d'utilisation et une ouverture vers l'extérieur. L'analyse des fichiers de données résoudrait le dernier problème. Pour le premier, il faudrait impérativement supprimer le relevé manuel des cotes, par exploitation directe des tracés effectués sur l'écran, en plan. Puis construire des routines capables d'assembler ces données, de les interpréter et enfin de les traduire en nombres. **Turbo Pascal**, avec ses routines « Turtle Graphic » devrait permettre un tel niveau d'automatisme, à condition de résoudre le problème relatif aux cercles et arcs de cercle : position du centre, rayon, parallélisme à l'un des trois plans, angles de départ et d'arrivée. Il serait même possible d'utiliser avec avantage une *souris*, ce qui apporterait un confort d'utilisation réellement « professionnel ».

Toutes sortes d'expérimentations sont envisageables. Cependant, le domaine du relief demeure encore au stade de la découverte, même si au niveau de la photographie ordinaire, ou de l'*holographie*, les résultats obtenus sont plus qu'encourageants. Il est donc difficile de prédire ce que sera l'avenir dans ce domaine. Quelques technologies, existantes ou à venir, pourraient le propulser à un premier plan industriel.

Par exemple il existe déjà des écrans plats, de type *L.C.D.*, utilisés pour les montres. Il est possible de les superposer, de façon à obtenir la visualisation soit d'un écran, soit du second. L'effet de ces *cristaux liquides* est assez directif sur le plan vertical et d'autre part leur extension à la couleur a déjà été réalisée. Tous les éléments existent donc, à l'heure actuelle, qui permettraient de présenter des images en **relief** et en couleur, sans port de lunettes spéciales. L'appareil pourrait même être miniaturisé, tout en possédant une capacité de stockage d'images impressionnante.

En attendant que le futur devienne l'actualité, ce qui ne saurait tarder, nous souhaitons bon courage au lecteur, avec les moyens mis d'ores et déjà à sa disposition.

MOTS-CLES

Voici la liste des mots-clés que vous avez pu rencontrer en marge, tout au long de ce livre, avec le renvoi à leur page d'appel.

A

B

C

D

E

F

G

H

I

L

M

O

P

R

S

T

V

TABLE DES MATIERES

3D et vrai relief
J.-J. Meyer
1e édition, 1er tirage



EDITIONS RADIO

Service lecteurs

(à retourner à Éditions Radio, 9, rue Jacob, 75006 Paris)

Pour nous permettre de vous proposer des ouvrages toujours meilleurs, nous souhaiterions recevoir vos critiques, appréciations et suggestions sur le présent livre :

____

____

____

____

Quels sont les ouvrages (thème, sujet, niveau) que vous souhaiteriez voir publier par notre société ?

____

____

____

____

Nous vous remercions de votre confiance et de votre coopération.

Éditions Radio

Je désire recevoir gratuitement et sans engagement (mettre une croix dans la case) :
☐ Votre catalogue général (Electronique professionnelle et grand public, Informatique, Hi-Fi, Vidéo)

Nom : ____________________ Prénom : ____________________

Adresse : ____

____

Secteur d'activité et fonction : ______________________________

____

CENTRES D'INTÉRÊTS

☐ Electronique professionnelle
☐ Electronique de loisirs
☐ Vidéo
☐ Hifi, CB...
☐ Micro-informatique professionnelle
☐ Micro-informatique de loisirs
☐ Autres : ....
....

Voir au dos "Correspondance Auteurs"

3D et vrai relief
J.-J. Meyer
1e édition, 1er tirage



Service lecteurs

Correspondance auteurs

(à retourner à Éditions Radio, 9, rue Jacob, 75006 Paris)

Pour toute demande d'éclaircissements techniques relatifs à ce livre, formulez ci-dessous vos questions, avec le maximum de précisions. Nous les transmettrons à l'auteur qui ne manquera pas de vous répondre directement :

Nom : ______________________ Prénom : ______________________

Adresse : ______________________

Avec, en plus, le listage d'un véritable progiciel original de tracé universel d'objets 3D.
(IBM PC – TURBO PASCAL)

3D et VRAI RELIEF

IMAGES DE SYNTHESE

Ce livre vous enseignera tout ce qu'il est nécessaire de savoir pour pénétrer dans le monde merveilleux du dessin tridimensionnel sur micro-ordinateur, trop souvent demeuré l'apanage des professionnels.

Il vous explique les principes et les règles des différentes techniques de création graphique en 3D et vrai relief accompagnés de programmes de démonstration écrits en BASIC Microsoft ou en TURBO PASCAL , langages disponibles sur la plupart des micro-ordinateurs.

Aucune connaissance informatique spécialisée ni connaissance mathematique approfondie ne sont nécessaires pour profiter pleinement de leurs possibilités graphiques extraordinaires, prêtes à « crever » l'écran, dans des domaines aussi variés que la mécanique, l'enseignement, la gestion, le dessin, l'animation et la recherche.

Cet ouvrage d'initiation, d'enseignement et de référence vous permet de comprendre et de concevoir des logiciels, écrits dans le langage de votre choix, capables de visualiser des objets ou des fonctions mathématiques, non seulement en trois dimensions, mais aussi en VRAI RELIEF, comme s'ils existaient réellement dans l'espace.

De plus, il vous fera découvrir le monde magique des images de synthèse et l'utilisation des courbes fractales en trois dimensions.

La CAO et la DAO ?... Mais c'est très simple !


9 782709 109901
ISBN 2 7091 0990 5
Code 154



ÉDITIONS RADIO

F 180/86/8

